P. 1877/91

5 1991

informatyka

Sieci Petriego
Komputeryzacja handlu detalicznego
Wyszukiwanie znaków
w rozpoznawaniu tekstu drukowanego

WYDAWNICTWO SIGMA NOT

Cena 12 000 zł

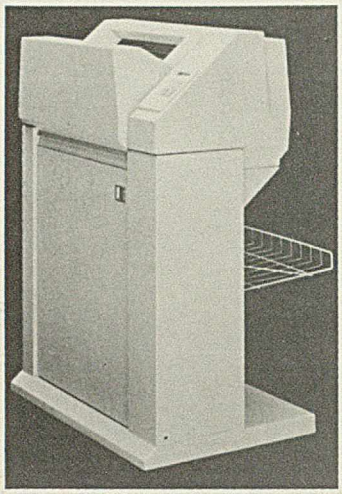
Supra

SUPRA SP. Z O.O.
ul. Trybunalska 54, 60-325 POZNAŃ
telefon: 674392, telefaks: 236498
teleks: 412966

AUTORYZOWANY DYSTRYBUTOR FIRMY MANNESMANN TALLY OFERUJE DRUKARKI HEAVY DUTY

MT 645
drukarka mozaikowo-wierszowa

- doskonała jakość druku
- niezawodność 7000 h MTBF
- duża szybkość 450 wierszy/min.
- cicha praca
- bufor 27 KB
- liczba kopii 1 + 5
- pełna grafika 240 (dpi)
- polskie litery (opcja)
- wewnętrzny program testujący
- dwa interfejsy automatycznie przełączane
- podwójny system traktorów

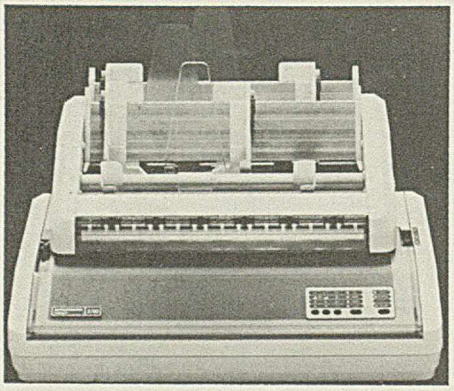


MT 340
drukarka mozaikowa (18 igłowa)

- doskonała jakość druku
- niezawodność
- szybkość 400 zn/sek.
- cicha praca
- polskie litery (opcja)
- druk kolorowy (opcja)

MT 230
drukarka mozaikowa (9, 18, 24 igłowa)

- doskonała jakość druku
- niezawodność
- szybkość 300 zn/sek.
- cicha praca
- polskie litery (opcja)
- druk kolorowy (opcja)



MANNESMANN TALLY

PRAWDOPODOBNIE NAJLEPSZE DRUKARKI NA ŚWIECIE!

0/2/91

informatyka

nr 5 1991
Maj
Rok wydania XXVI

Miesięcznik
ISSN 0542-9951
INDEKS 36124

POLITECHNIKA ŁÓDZKA Biblioteka Główna

P. 1877/91

KOLEGIUM REDAKCYJNE:

mgr Jarosław DEMINET
dr inż. Wacław ISZKOWSKI
mgr Teresa JABŁOŃSKA
(sekretarz redakcji)
Władysław KLEPACZ
(redaktor naczelny)
dr inż. Wojciech MOKRZYCKI
mgr inż. Jan RYŻKO
mgr Hanna WŁODARSKA

PRZEWODNICZĄCY
RADY PROGRAMOWEJ:

Prof. dr hab.
Juliusz Lech KULIKOWSKI

WYDAWCA:
Wydawnictw Czasopism i Książek Technicznych SIGMA NOT Spółka z o.o.
ul. Biała 4
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja:
01-552 Warszawa,
Pl. Inwalidów 10, p. 104, 105
tel. 39-14-34

Materiałów nie zamówionych redakcja nie zwraca

W sprawach ogłoszeń prosimy zwracać się bezpośrednio do Redakcji lub Działu Reklamy i Marketingu 00-950 Warszawa ul. Biała 4 telefon: 20-31-24 telefaks: 20-31-16 teleks: 814550

W numerze:

	Strona
Analizator Uogólnionych Stochastycznych Sieci Petriego – *Kamil Skudlarski*	3
Systematyczne testowanie programów – zalety i wady (2) – *Dorota Kuchta*	8
Komputeryzacja handlu detalicznego przy użyciu kas systemowych – *Andrzej Górecki*	11
Problemy wyszukiwania znaków w rozpoznawaniu tekstu drukowanego – *Cezary Grzegorz Dołęga*	16
Z kraju	
X Jubileuszowy INFOGRYF '90. Informatyka w zarządzaniu przedsiębiorstwem – rozwój metod i systemów informatyki stosowanej w zarządzaniu – *J. Orylska*	20
Wyniki Konkursu prac magisterskich – *Zbigniew Szpunar*	22
Czwarta Szkoła Wiosenna PTI – *Zdzisław Szyjewski*	22
unix	
Procesy, procesy... (w shellu Bourne'a) – *Jan Walasek*	24
O standardzie ANSI języka C (2) – *Władysław Majerski*	26
zBITki	
Felieton dla studentów informatyki – *Wacław Iszkowski*	III s. okł.

W najbliższych numerach:

- **Wojciech Trojniar przedstawia szerokie spektrum dostępnych w kraju systemów czasu rzeczywistego.**
- **Krzysztof Nałęcki charakteryzując system operacyjny czasu rzeczywistego, ujawnia jego ograniczenia i wykazuje, że wymagań tych nie może spełniać system typu UNIX.**
- **Paweł Gizbert-Studnicki opisuje środowisko do projektowania prostych systemów czasu rzeczywistego.**

Warunki prenumeraty

Przyjęcie prenumeraty – wyłącznie na podstawie dokonanej wpłaty na drukach dostarczanych dotychczasowym prenumeratorom przez Wydawnictwo, lub nowym – po uprzednim zgłoszeniu zapotrzebowania (pisemnie lub telefonicznie) w Zakładzie Kolportażu Wydawnictwa.

Blankiet wpłaty – powinien zawierać następujące informacje: dokładna nazwa i adres (z kodem pocztowym) zamawiającego, tytuły zamawianych czasopism, ich liczbę i okres prenumeraty.

Wpłata – zgodnie z podanymi cenami należy dokonać w banku lub w UPT na konto podane na naszym blankiecie, tj:
Państwowy Bank Kredytowy III O/Warszawa nr: 370015-1573-139-11

Prenumeratorzy zbiorowi – osoby prawne obowiązują blankiety „Wpłata-Zamówienie". Cena normalna.

Prenumeratorzy indywidualni – osoby fizyczne obowiązują blankiety typu przekazy dla wpłat na rachunki bankowe. Cena normalna.

Prenumerata ulgowa – zgodnie z podaną ceną ulgową przysługuje wyłącznie osobom fizycznym, będącym członkami SNT, studentom i uczniom szkół zawodowych. Uczniowie szkół ogólnokształcących mogą zamówić w prenumeracie ulgowej tylko miesięcznik „Aura"

Uwaga! w podanym okresie prenumeraty można zamówić tylko po jednym egzemplarzu z każdego tytułu.

Prenumerata ze zleceniem wysyłki za granicę – cena prenumeraty ze zleceniem wysyłki za granicę jest dwukrotnie wyższa od ceny normalnej.

Należy podać dokładny adres odbiorcy za granicą.

Terminy przyjmowania prenumeraty:
– do 10 listopada na I, II, III, IV kwartał następnego roku
– do 28 lutego na II, III, IV kwartał br.
– do 31 maja na III i IV kwartał br.
– do 31 sierpnia na IV kwartał br.

Zmiany w prenumeracie, np. zmiana liczby tytułów, liczby egzemplarzy, rezygnacja z prenumeraty, można zgłaszać tylko w podanych terminach z mocą obowiązującą od następnego kwartału.

Egzemplarze archiwalne (z lat ubiegłych)
Można nabyć za gotówkę w Klubie Prasy Technicznej, Warszawa, ul. Mazowiecka 12 (tel. 26-80-16) lub zamówić pisemnie w Zakładzie Kolportażu, Dział Handlowy, 00-950 Warszawa, skr. poczt. 1004 (tel. 40-37-31), na rachunek lub za zaliczeniem pocztowym.

Informacji o prenumeracie udziela: Zakład Kolportażu Wydawnictwa SIGMA-NOT Spółka z o.o., 00-716 Warszawa, ul. Bartycka 20, skr. 1004. Telefony: 40-00-21 wewn. 293, 295, 299 lub 40-30-86, 40-35-89.

Wstępna cena jednego egzemplarza na 1991 rok: normalna – 12 000 zł ulgowa – 8400 zł

Wartość prenumeraty (w zł):
Normalna: kwartalna – 36 000, półroczna 72 000, roczna 144 000
Ulgowa: kwartalna – 25 200, półroczna 50 400, roczna 100 800

Uwaga: W przypadku zmiany cen w okresie objętym prenumeratą, prenumeratorzy zobowiązani są do dopłaty różnicy cen.

Skład i druk: Drukarnia SIGMA NOT Sp. z o.o. z. 117/91 n. 3650 egz.

POZNAJ SIŁĘ NASZYCH SYSTEMÓW INFORMATYCZNYCH!

JUNISOFTEX Sp. z o.o.
44-100 Gliwice ul. Konstytucji 11
Tel. 31-75-10, 31-90-81 do 88 wewn. 250, 272, telefaks: 31-75-10, teleks 036233

Już od 3 lat oferuje kompleksową komputeryzację przedsiębiorstw, a zwłaszcza:

- profesjonalne nowoczesne systemy informatyczne oparte na kompilatorze języka Turbo Pascal 5.5 firmy BORLAND (typu multiusers), pracujące w sieciach NOVELL 2.15 i 3.1 i wykorzystujące interfejs BTRIEVE amerykańskiej firmy SoftCraft – najefektywniejszy i najszybszy system obsługi dużych i bardzo dużych baz danych (do 4 miliardów rekordów w pojedynczej bazie danych).

Polecamy szczególnie systemy zintegrowane

- wielodostępny system finansowo-księgowy wg trójstopniowej analityki w podstawowym układzie: xxx-xxxx-xxxx-xxxxxx z możliwością graficznej prezentacji wyników,
- wielodostępny system gospodarki materiałowej wykorzystujący w strukturze indeksowej SWW i KTM z automatycznym wyliczaniem cyfry kontrolnej (długość indeksu materiałowego 13 + 3 cyfry),
- wielodostępny system gospodarki wyrobami gotowymi wykorzystujący w strukturze indeksowej SWW i KTM,
- wielodostępny system gospodarki przedmiotami nietrwałymi wykorzystujący w strukturze indeksowej SWW i KTM,
- wielodostępny system środków trwałych wykorzystujący klasyfikację środków trwałych GUS,
- wielodostępny system fakturowania sprzedaży,
- wielodostępny system kadrowo-płacowy.

Wszystkie oferowane systemy są zintegrowane ze sobą zgodnie z zasadami rachunkowości księgowej przedstawionymi w miesięczniku RACHUNKOWOŚĆ nr 9, 1990.

Oferujemy również:

- sprzęt komputerowy (sprawdzony w eksploatacji) firm:
  – ALR (Advanced Logic Research) i Storage Dimensions z USA,
  – AZTECH, WEARNES TECHNOLOGY, NAC i inne z Singapuru,
- dostawę i instalację licencjonowanej sieci Novell 2.15 i 3.1,
- usługi w zakresie opracowywania zakładowych indeksów materiałowych wg klasyfikacji GUS (SWW i KTM).

Organizujemy:

- szkolenia i prezentację systemów u naszych kontrahentów.

Gwarantujemy:

- najwyższą jakość naszych systemów i usług,
- najszybsze w Polsce wdrożenia systemów (od 1 do 3 miesięcy).

W przypadku niedotrzymania przez nas terminów zwracamy 25% poniesionych nakładów za każdy miesiąc zwłoki. Dla klientów zamawiających kompleksowe usługi – znaczne zniżki cenowe.

Poszukujemy niezawodnych

- pośredników i dealer'ów naszych systemów informatycznych (oferujemy bardzo atrakcyjne warunki współpracy),
- dostawców i pośredników dobrego i taniego sprzętu komputerowego.

Z siły naszych systemów skorzystały dotąd następujące firmy:

Huta ŁAZISKA w Łaziskach Górnych	tel. 24-14-06
Walcownie Metali w Czechowicach-Dziedzicach	tel. 52-35-1
Centrala Zaopatrzenia Górnictwa w Bytomiu	tel. 81-92-91
Mikołowska Fabryka Transformatorów w Mikołowie	tel. 26-25-41
BPiKD PREFAMET-ZREMB w Gliwicach	tel. 38-20-51
BPiKD METALCHEM w Gliwicach	tel. 31-56-33
Instytut Informatyki Teoretycznej i Stosowanej PAN w Gliwicach	tel. 31-71-50
Instytut Spawalnictwa w Gliwicach	tel 31-00-11
PP ENERGOROZRUCH w Gliwicach	tel. 37-66-25
Fabryka Palenisk Mechanicznych w Mikołowie	tel. 26-20-02
Zakłady Mechaniczne Górnictwa i Energetyki WIROMET w Mikołowie	tel. 26-23-46
Przędzalnia Bawełny PRZYJAŹŃ w Zawierciu	tel. 22-39
PRT TERMOIZOLACJA w Zabrzu	tel. 71-40-21
Przedsiębiorstwo Usług Informatycznych ETOM w Mikołowie	tel. 57-10-50
Zakład Informatyki i Techniki Komputerowej MEGA w Gliwicach	tel. 31-52-42
Mikołowska Fabryka Maszyn Górniczych MIFAMA w Mikołowie	tel. 26-22-66
Zakłady Produkcji Urządzeń Mechanicznych ELWO w Pszczynie	tel. 30-61
Zakłady Sprzętu Sieciowego i Elektroinstalacyjnego POLAM w Katowicach	tel. 52-80-31
Urząd Miasta w Mikołowie	tel. 26-23-90
Zakłady Elelktromaszynowe CELMA w Cieszynie	tel. 21-58-1
PSS SPOŁEM w Zawierciu	tel. 21-46-17

Chcesz mieć pełną gwarancję i niezawodne efekty komputeryzacji swojego przedsiębiorstwa – współpracuj z firmą JUNISOFTEX!

0/1/91

Sieci Petriego przyciągają ostatnio na świecie uwagę coraz szerszego grona naukowców, inżynierów i ekonomistów. Zaczynają znajdować zastosowanie w najprzeróżniejszych dziedzinach, głównie tam gdzie mamy do czynienia z wszelkiego rodzaju analizą: kosztów, efektywności, wydajności, współdziałania i dopasowania. Teoria sieci Petriego rozwijana jest również w kilku ośrodkach w Polsce.

KAMIL SKUDLARSKI
Centrum Obliczeniowe
Politechnika Wrocławska

Analizator Uogólnionych Stochastycznych Sieci Petriego

Opisywane już na łamach INFORMATYKI [2, 3] sieci Petriego są doskonałym narzędziem do modelowania wszelkich zjawisk zachodzących współbieżnie. Podstawowe wiadomości o nich można znaleźć w książkach [7, 8, 10]. W poniższym artykule przedstawiono pakiet programów służący do analizy Uogólnionych Stochastycznych Sieci Petriego (USSP). Sieci te [5] są uogólnieniem Stochastycznych Sieci Petriego zdefiniowanych przez Molloya [6]. Oprócz przejść, których czas palenia zadany jest rozkładem wykładniczym, mają one również tak zwane przejścia natychmiastowe, palące się w pierwszej kolejności. W pracy [5] podany jest algorytm redukcji ze zbioru stanów osiągalności sieci, stanów, w których mogą palić się przejścia natychmiastowe (stanów chwilowych).

Do tej pory powstało kilka programów służących do numerycznej analizy sieci Petriego. Pierwszy zintegrowany i łatwy w użyciu pakiet programów [8] przeznaczony jest na komputery VAX 11/780. Opisany tutaj system ASP jest przeznaczony do pracy na komputerach klasy IBM PC. Napisany został w języku Turbo Pascal i ma budowę modułową. Umożliwia to jego dalszą rozbudowę w miarę rozwoju teorii sieci Petriego. Obecnie pakiet oprócz dokładnego rozwiązania sieci umożliwia również dwa bardzo efektywne sposoby analizy przybliżonej (dające się jednak zastosować tylko dla wąskiej klasy USSP). Przy wyborze algorytmu optymalizującego starano się raczej minimalizować złożoność pamięciową niż czasową. W rezultacie uzyskano możliwość rozwiązywania stosunkowo dużych sieci (każdej o przestrzeni stanów do 290, a niekiedy nawet do 2000), przy czasie pracy (dla IBM PC/XT 8 MHz) dochodzącym w niektórych przypadkach do kilku godzin.

Możliwości pakietu ASP

Przedstawiony system przygotowany jest w dwóch wersjach, różniących się możliwościami i obsługą. Pierwsza (MASP – Mini Analizator Sieci Petriego), jest przeznaczona do obsługi niedużych sieci (generujących do 290 stanów). Wersja druga (ASP – Analizator Sieci Petriego) umożliwia analizę znacznie większych sieci (niekiedy mogą one generować nawet do 2000 stanów). Program MASP jest prostszy w obsłudze i w związku z tym jest przeznaczony dla początkujących użytkowników. Może on jednakże obliczać mniejszą liczbę parametrów sieci i operuje na węższej ich klasie. W dalszej części artykułu przez skrót ASP będziemy rozumieli oba programy, chyba, że będzie to inaczej wyraźnie zaznaczone.

ASP (nie MASP) służy do wszechstronnej analizy (opisanych w pracy [6]) uogólnionych stochastycznych sieci Petriego, w których prawdopodobieństwa palenia przejść natychmiastowych mogą zależeć od znakowania, a intensywności przejść czasowych mogą być mnożone przez mnożniki, również zależne od znakowania. Za jego pomocą można obliczyć wszystkie wyżej wymienione parametry. Przewidziano również możliwość samodzielnego definiowania obliczanych wielkości. Wyniki obliczeń mogą być wyświetlane na monitorze lub drukowane na drukarce. Formaty wydruków można zdefiniować indywidualnie lub skorzystać z postaci standardowej.

MASP służy do analizy sieci, w których intensywności przejść czasowych i prawdopodobieństwa przejść natychmiastowych nie zależą od znakowania.

Program ASP (nie MASP) może w jednej serii automatycznie powtarzać zadane przez użytkownika obliczenia, zmieniając za każdym razem dane wejściowe (umożliwia to wygodne badanie zależności wybranej wielkości od zmian jednego z parametrów sieci). Przykładowo, można obliczać wpływ intensywności palenia przejścia na średnie liczby znaczników w miejscach, prawdopodobieństwa stanów, czasu cyklu itp. Do pakietu jest dołączony pomocniczy program GRAFINT, służący do interpretacji graficznej uzyskanych wyników (rysuje on wykresy zadanych zależności, gdy obliczenia są powtarzane w pętli przy zmieniających się parametrach). ASP może wykonać obliczenia dla każdej sieci, której przestrzeń stanów uchwytnych nie przekracza 290 stanów. Dla niektórych sieci, stosując metodę Gaussa-Siedla, można obliczyć parametry sieci nawet wtedy, gdy jej przestrzeń stanów dochodzi do 2000.

Organizacja pakietu

Pakiet ASP napisany został na komputerze IBM PC/XT w języku Pascal. Początkowo korzystano z kompilatora Turbo Pascal 3.0, a następnie programy przystosowano do wersji Turbo Pascal 4.0. Struktura programów ASP i MASP (podział na moduły i wykonywane przez nie funkcje) jest identyczna. Proces analizy sieci Petriego składa się z kilku etapów wykonywanych przez przeznaczone do tego celu moduły.

Cały pakiet składa się z dwunastu niezależnych modułów. Niektóre z nich rozwiązują poszczególne etapy analizy sieci Petriego, inne pełnią tylko role pomocnicze i usługowe względem pozostałych. Poniżej przedstawiono charakterystykę roli, jaką w systemie pełnią poszczególne moduły. Główny moduł, zarządzający pracą całego systemu, nosi nazwę ASP (MASP w wersji uproszczonej). Moduły przekazują sobie niektóre dane za pośrednictwem dysku, tworząc na nim w kartotece o nazwie TYM pewne zbiory pomocnicze.

Analizator danych wejściowych

W skład tego bloku wchodzą moduły *PETDAN* oraz *CZYTPAR*. Moduł *PETDAN* zawiera wszelkie dane dotyczące rozwiązywanej sieci. Na jego podstawie wszystkie pozostałe moduły budują niezbędne struktury danych (przydzielają i zwalniają odpowiedniej wielkości obszary pamięci) oraz sterują przebiegiem obliczeń. Moduł *CZATPAR* przetwarza wstępne dane z *PETDAN* i udostępnia je pozostałym modułom.

Generator stanów osiągalności

Przestrzeń stanów sieci generowana jest przez moduł *STANOS*. Jednocześnie liczone są prawdopodobieństwa przejść między stanami i eliminowane są stany chwilowe. Danymi wejściowymi do tego modułu są wyrażenia opisujące mnożniki intensywności przejść czasowych, wyrażenia na prawdopodobieństwa przejść natychmiastowych, definicja połączeń sieci, znakowanie początkowe, początkowe intensywności przejść czasowych. *STANOS* produkuje zbiór *PETDAN.GRF* zawierający opis grafu przejść między stanami sieci i prawdopodobieństwa przejść. Oblicza również czasy pobytu w stanach. Moduł *PRAWDST* zawiera procedury służące do obliczania prawdopodobieństw odpalenia przejść (zarówno czasowych jak i natychmiastowych).

W efekcie pracy tego modułu zostanie znaleziona przestrzeń stanów widocznych sieci Petriego. Stany chwilowe nie są przez program generowane.

Moduły rozwiązujące układ równań

Program ASP może rozwiązywać układy równań dwoma metodami – za pomocą eliminacji Gaussa (rozwiązanie dokładne) oraz metodą iteracyjną Gaussa-Seidla (rozwiązanie przybliżone, jednakże błąd przybliżenia może być nawet mniejszy niż dokładność operacji wykonywanych przez komputer). Pierwszą metodę stosuje moduł *GAUSS_EL* a drugą *GAUSS_SE*. O wyborze metody decyduje użytkownik. Moduły *MAC_UP* i *MAC_PEL* definiują format zapisu macierzy w pamięci komputera, zawierają również procedury do zapisu i odczytu elementów tej macierzy. Umożliwia to łatwą zmianę sposobu pamiętania macierzy przez zdefiniowanie odpowiedniego modułu udostępniającego na zewnątrz procedury zapisu i odczytu tej macierzy. Moduł *MAC_PEL* zapisuje wszystkie elementy macierzy w pamięci operacyjnej i wykorzystywany jest przez moduł *GAUSS_EL*. Za jego pomocą można rozwiązać układ do 290 równań liniowych. *MAC_UP* zapisuje w pamięci operacyjnej komputera tylko elementy niezerowe macierzy i wykorzystywany jest przez moduł *GAUSS_EL*. Wymiar układu równań, jaki można w ten sposób rozwiązać zależy od liczby elementów niezerowych w jego macierzy. Niektóre układy równań nie dają się rozwiązać metodą Gaussa-Seidla, gdyż otrzymywany w niej szereg nie jest zbieżny. Niekiedy zmiana numeracji miejsc lub przejść w sieci może spowodować, że rozbieżny szereg stanie się zbieżnym (a zbieżny bardziej zbieżnym). Niestety nie stwierdzono w tej dziedzinie żadnych regularności i przenumerowanie należy przeprowadzać metodą prób i błędów. Możliwe jest również korzystanie przez moduł *GAUSS_EL* z *MAC_UP* i przez moduł *GAUSS_SE* z *MAC_PEL*, jednakże nie wydaje się to być uzasadnione.

Moduł wykonujący obliczenia parametrów sieci

Zadaniem tego modułu jest wykonanie wszelkich obliczeń końcowych, takich jak obliczenie:

- gęstości znaczników w miejscach,
- czasu cyklu sieci,
- procentu czasu spędzonego w danym stanie,
- wszelkich zdefiniowanych przez użytkowników obliczeń.

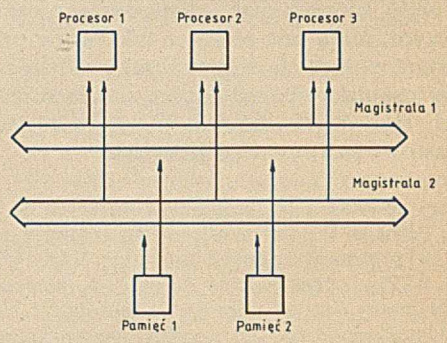
Moduł ten nazywa się *OBL_PAR* i fizycznie jest częścią modułu głównego ASP (nie jest on zaimplementowany w żadnym osobnym zbiorze dyskowym). Korzysta on z prawdopodobieństw granicznych (wyliczonych przez moduł opisany w poprzednim punkcie) oraz z czasów pobytu w poszczególnych stanach (wyliczonych przez wyżej opisany moduł *STAN_OS*). Użytkownik definiując własne rodzaje obliczeń ingeruje pośrednio w postać opisywanego modułu (nie dotyczy to wersji ASP).

Przykłady zastosowania pakietu

Poniżej przedstawiono dwa przykłady obliczeń wykonanych za pomocą sieci Petriego. Odzwierciedlają one najważniejsze kierunki zastosowań tych sieci. Wszystkie przykłady rozwiązane za pomocą programu ASP wykorzystują czasowe uogólnione Sieci Petriego.

Przykład 1. Model systemu wieloprocesorowego

Rozpatrzmy przedstawiony na rys. 1, a omawiany już w pracach [1, 5, 6] układ złożony z kilku procesorów, kilku pamięci wspólnych oraz dwóch magistral. Sieć Petriego opisującą ten system przedstawiono na rys. 2. Zależność gęstości znaczników w miejscu P_1 od intensywności palenia przejścia t_5 (czyli średniej liczby procesorów pracujących aktywnie od częstotliwości żądań dostępu do wspólnej pamięci), obliczona dla trzech pamięci, czterech procesorów i dwóch magistral przedstawiono na rys. 3. Rysunek 4 przedstawia zależność średniej



Rys. 1. Architektura systemu komputerowego z trzema procesorami, dwoma pamięciami zewnętrznymi i dwoma magistralami

Metoda analizy stochastycznych sieci Petriego

Opisywany system korzysta z najczęściej stosowanej metody sprowadzającej sieć Petriego do łańcucha Markowa. Można w niej wyróżnić kilka etapów.

Generacja przestrzeni stanów sieci

Znalezienie wszelkich możliwych rozkładów znaczników w miejscach sieci. Dla stanu początkowego (znakowania początkowego M_0) odpalamy wszystkie możliwe do odpalenia przejścia i sprawdzamy, czy pojawiające się w ten sposób stany wystąpiły już wcześniej. Dla stanów, które pojawiły się po raz pierwszy (nie wystąpiły wcześniej) powtarzamy powyższą procedurę. Gdy przestrzeń stanów jest ograniczona (liczba możliwych do wygenerowania znakowań jest ograniczona), postępowanie to zakończy się.

KONSTRUKCJA ŁAŃCUCHA MARKOWA

Następnym krokiem analizy sieci Petriego jest znalezienie prawdopodobieństw przejść pomiędzy stanami. Powiedzmy, że interesuje nas przejście ze stanu S_i do S_k. Jeżeli przejście przeprowadzające sieć ze stanu S_i do S_k jest natychmiastowe, to prawdopodobieństwo jego palenia jest właśnie prawdopodobieństwem przejścia ze stanu S_i do S_k. Gdy przejście to jest czasowe i ma intensywność $\lambda_{l(i,k)}$, to szukane prawdopodobieństwo wyraża się wzorem:

$$\frac{\lambda_{l(i,k)}}{\sum_j \lambda_{l(i,j)}} \quad \text{gdzie} \quad \sum_j \lambda_{l(i,j)}$$

jest sumą intensywności wszystkich przejść możliwych do odpalenia w stanie $S_i \cdot l(i,j)$, i jest funkcją zwracającą indeks przejścia przeprowadzającego sieć ze stanu S_i do S_j.

Obliczanie prawdopodobieństw stanów w warunkach równowagi

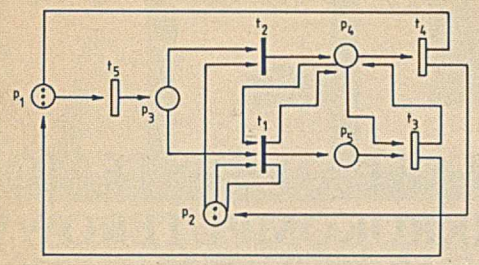
Gdy mamy do czynienia z siecią, w której nie ma przejść natychmiastowych (wszystkie stany są widoczne), to budujemy macierz intensywności Q taką, że

$$q_{i,j} = \lambda_{l(i,j)} \qquad \text{dla } i <> j \quad \text{a}$$

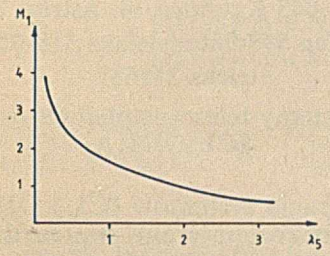
$$q_{i,j} = -\sum_{j \neq i} \lambda_{l(i,j)}$$

(gdzie $\lambda_{l(i,j)}$ jest intensywnością przejścia ze stanu S_i do stanu S_j). Niech P_i będzie prawdopodobieństwem granicznym (w warunkach równowagi) osiągnięcia stanu S_i. Prawdopodobieństwa P_i spełniają warunek $\sum_i P_i = 1$. Po wstawieniu tego równania zamiast jednego z równań w układzie macierzowym $P * Q = 0$, otrzymamy układ równań, którego rozwiązaniem są prawdopodobieństwa graniczne.

liczby aktywnie pracujących procesorów od ich ogólnej liczby w systemie. Przykład ten jest powtórzeniem obliczeń wykonanych przez M.A. Marsan, G. Conte, G. Balbo [5] i wykazuje całkowitą zgodność obu wyników.



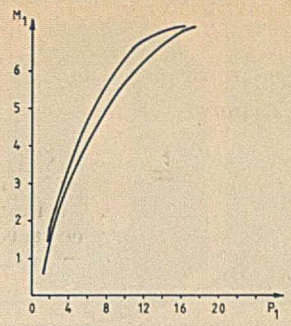
Rys. 2. Sieć Petriego modelująca system komputerowy z rys. 1



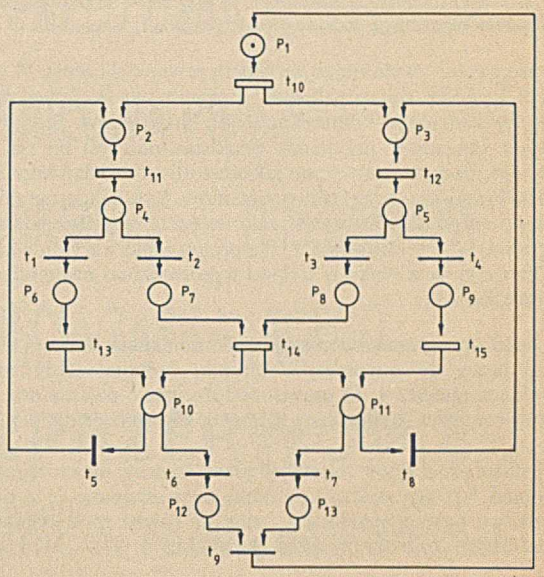
Rys. 3. Wyliczona zależność gęstości znaczników w miejscu P_1 od intensywności palenia przejścia t_5 (czyli średniej liczby procesorów pracujących aktywnie od częstotliwości żądań dostępu do wspólnej pamięci) obliczona dla trzech pamięci, czterech procesorów i dwóch magistral

Przykład 2. Modele programów współbieżnych

Przykład ten opiera się na przedstawionej w pracy Magotta [4] koncepcji wykorzystania sieci Petriego do analizy oprogramowania współbieżnego. W przykładzie modeluje się zachowanie dwóch identy-



Rys. 4. Zależność średniej liczby aktywnie pracujących procesorów od liczby wszystkich procesorów włączonych do systemu, dla dwóch magistral, trzech i dziesięciu pamięci



Rys. 5. Sieć Petriego modelująca komunikujące się ze sobą programy

Gdy w sieci występują przejścia natychmiastowe, to macierz Q budujemy eliminując jednocześnie stany chwilowe [6]. Dalsze postępowanie jest takie same, jak w przypadku sieci bez przejść natychmiastowych. W ten sposób wyliczamy prawdopodobieństwa stanów uchwytnych sieci. Dla stanów chwilowych prawdopodobieństwa te wynoszą zero.

Obliczenia parametrów sieci

Po znalezieniu prawdopodobieństw granicznych można wyliczyć następujące parametry sieci:

- średni czas C_i pobytu sieci w zadanym stanie uchwytnym S_i równy

$$C_i = \frac{1}{\sum_{j \in W_i} \lambda_{l(i,j)}}$$

l, i, j – jest funkcją zwracającą indeks przejścia przeprowadzającego sieć ze stanu S_i do S_j,
W_i – jest zbiorem numerów przejść możliwych do odpalenia w stanie S_i;

- średni ułamek czasu K_i spędzony przez sieć w stanie S_i równy

$$K_i = \frac{P_i * C_i}{\sum_{j \in S} P_j * C_j}$$

S jest zbiorem numerów stanów osiągalności.

- średnia liczba znaczników M_i w miejscu i-tym równa

$$M_i = \sum_{j \in S} m_{i,j} * K_j$$

$m_{i,j}$ jest liczbą znaczników w miejscu i-tym, gdy sieć znajduje się w stanie S_j;

- prawdopodobieństwo, że w miejscu P_i znajduje się k znaczników równe

$$\sum_{j \in S(i,k)} P_j * K_j$$

$S(i,k)$ jest zbiorem stanów S_j, dla których $m_{i,j} = k$;

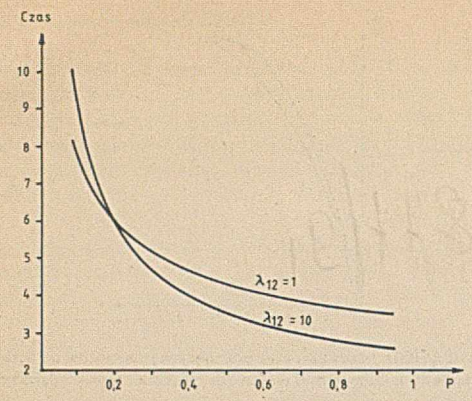
- średnia liczba przejść przez stan S_i pomiędzy dwoma kolejnymi przejściami przez stan

$$S_j N_{i,j} = \frac{P_i}{P_j};$$

- średni czas powrotu do stanu S_i po wyjściu ze stanu S_i (nazywamy go dalej czasem cyklu sieci)

$$T_i = \sum_{j \in S} N_{i,j} * C_j$$

Oczywiście można wyliczać dowolne inne wielkości, które dają się wyrazić za pomocą wyżej podanych parametrów.



Rys. 6. Zależność czasu pracy komunikujących się programów od częstotliwości z jaką jeden z nich żąda komunikacji, dla intensywności przejścia t_{12} równych 1 i 10

cznych programów, wykonujących w pętli pewne obliczenia. W obrębie tych pętli znajduje się rozgałęzienie programów. W jednej z gałęzi znajduje się instrukcja komunikacji obu programów. Sieć Petriego modulująca omawiany przypadek przedstawiona jest na rys. 5. Za pomocą tego modelu sprawdzono jak częstotliwość żądań komunikacji wpływa na łączny czas pracy obu programów. Rys. 6 ilustruje zależność czasu pracy całego układu (czas do zakończenia pracy dłużej działającego programu) od częstotliwości, z jaką program pierwszy żąda komunikacji. Dla uniknięcia blokady układu wprowadzono następujące dwa dodatkowe warunki:

1. Gdy jeden z programów oczekuje na komunikację, to drugi nie może przerwać pracy do momentu obsłużenia żądania. (Gdy znacznik znajduje się w miejscu p_7 to prawdopodobieństwo palenia przejścia t_7 staje się równe zeru. Analogiczna zależność obowiązuje dla przejścia t_6 i miejsca p_8).
2. Gdy jeden z programów zakończył pracę, to drugi nie może już żądać komunikacji. Np. gdy znacznik znajduje się w miejscu p_{12}, to prawdopodobieństwo palenia przejścia t_3 staje się równe zeru. Analogiczna zależność obowiązuje dla przejścia t_2 i miejsca p_{13}.

* * *

Do praktycznego wykorzystania sieci Petriego, a także do weryfikacji niektórych założeń teoretycznych, niezbędne jest specjalistyczne oprogramowanie. Niestety ze względu na wymagania sprzętowe, oprogramowanie powstające za granicą jest w Polsce mało przydatne. System ASP nadaje się do szerokiego wykorzystania w warunkach krajowych. Umożliwia on analizowanie nie tylko standardowych sieci Petriego, ale i wielu ich rozszerzeń. Jest on jednocześnie przystosowany do łatwej rozbudowy i wprowadzania modyfikacji. Biorąc pod uwagę złożoność obliczeniową problemu oraz moc obliczeniową zastosowanego komputera (XT, AT), opisany program może analizować sieci całkiem pokaźnych rozmiarów. Szczegółowy opis działania i instrukcję obsługi ASP można znaleźć w pracy [9].

Idea opracowania programu ASP powstała w Instytucie Cybernetyki Technicznej Politechniki Wrocławskiej, od pewnego czasu jest on z powodzeniem używany. Pierwszą wersję programu wykorzystuje również od 1988 r. Instytut Matematyki Stosowanej Wyższej Szkoły Pedagogicznej w Rzeszowie.

LITERATURA

[1] Ciola G.: Software Package for the Analysis of Generalized Stochastic Petri Net Models. In Proc. Int. Workshop on Timed Petri Nets, IEEE, Torino, Italy, July 1985
[2] Gondzio M: Sieci Petriego – definicja i analiza własności sieci. INFORMATYKA, styczeń 1987
[3] Gondzio M.: Sieci Petriego – przekształcenia i rodzaje sieci. INFORMATYKA, luty 1987
[4] Magott J.: Performance evaluation of concurrent systems
[5] Marsan M.A., Conte G., Balbo G.: A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocesor Systems. ACM Transactions on Computer Systems, Vol. 2, May 1984
[6] Molloy M.K.: On the Integration of Delay and Throughput Measures in Distributed Processing Models. Ph.D. Thesis, UCLA, Los Angeles, 1981
[7] Peterson J.L.: Petri Net Theory and the Modelling of Systems. Prentice Hall, Englewood Cliffs, N. J. 1981
[8] Reisig W.: Sieci Petriego. WNT 1988
[9] Skudlarski K.: Analizator Uogólnionych Stochastycznych Sieci Petriego. Centrum Obliczeniowe Politechniki Wrocławskiej, Raport Serii PRE nr 6, 1990
[10] Starke P.H.: Sieci Petri. PWN, 1987.

Microcomp

ZAKŁAD SYSTEMÓW MIKROKOMPUTEROWYCH

MiCOMP

40–045 Katowice, ul. Astrów 7
telefon 513-086, telefaks 518-628,
teleks 315687

Systemy teletransmisji danych ICL/ODRA

Integracja systemów ICL i ODRA z mikrokomputerami standardu PC i systemami NOVELL, UNIX SCO

procesor sieci rozległej WAN MiCOMP 8075 (emulacja ICL 7503)

most NOVELL – ICL/ODRA NetCROSS-FXBM
program teletransmisji danych MiCROS-FXBM (PC, rozproszone bazy danych, wersja sieciowa – NOVELL),

skaner MPXSCAN-8007,

adaptery, modemy, testery, emulatory terminali

karty BSC do połączeń komputerów:
PC/XT/AT z PC/XT/AT
PC/XT/AT z IBM/370, R32, R34
z emulatorami IBM 3270/3780
PC/XT/AT z ICL

karty kompatybilne z IRMA do połączeń
PC/XT/AT z IBM 370
lokalnie przez IBM 3274

karty SNA/SDLC

konwertery o szybkościach do 19200 bps do zdalnego przyłączania terminali

przystawki teleksowe współpracujące z mikrokomputerami PC/XT/AT

Instalacje w największych systemach na terenie całego kraju!

0/23/90

MEDCOM

CHCESZ ZABEZPIECZYĆ BAZY DANYCH PRZED UTRATĄ INFORMACJI?

CHCESZ ZAPEWNIĆ PRAWIDŁOWĄ PRACĘ

KOMPUTERA * CENTRALI TELEFONICZNEJ * TELEFAKSÓW

W WARUNKACH ZANIKU NAPIĘCIA ZASILAJĄCEGO

kup ZASILACZE BEZPRZERWOWE

naszej produkcji
UPS – 120 (abonenckie centrale telefoniczne i telefaksy)
UPS – 450
UPS – 850
UPS – 1450

produkcji
AMERICAN POWER CONVERSION:

UPS – 110
UPS – 310
UPS – 400
UPS – 600
UPS – 800
UPS – 1200

produkcji BEST POWER TECHNOLOGY ON – LINE:
500 VA – 18 kVA

Sprzedajemy również

* AKUMULATORY BEZOBSŁUGOWE, HERMETYCZNE FIRMY PANASONIC do zasilania urządzeń alarmowych, jedyne dopuszczone do używania w pomieszczeniach zamkniętych,
* GENERATORY PRĄDOTWÓRCZE FIRMY ENDRESS O MOCY OD 1 kVA DO 350 kVA

Twoje problemy w dziedzinie zasilania awaryjnego rozwiążą specjaliści z firmy MEDCOM.

BIURO
Al. Ujazdowskie 26/39
00-478 Warszawa
tel. 628-93-57, teleks 817060

ZAKŁAD PRODUKCYJNO-USŁUGOWY
ul. Bokserska 53
02-690 Warszawa

0/20/90

DOROTA KUCHTA
Instytut Organizacji i Zarządzania
Politechnika Wrocławska

Systematyczne testowanie programów zalety i wady (2)

W pierwszej części artykułu [3] omówiono różne rodzaje metod testowania poprawności programów komputerowych. Uczyniono to w sposób popularny i poglądowy, przez prezentacje funkcjonowania tych metod na przykładach prostych programów komputerowych. W podobny sposób zostaną teraz omówione wady i zalety wspomnianych metod, a także propozycje ich usprawnienia, pod względem zdolności do wykrywania błędów w programach komputerowych.

Do ilustracji dyskusji zostaną wykorzystane m.in. zadania i programy przedstawione w pierwszej części artykułu (przy odsyłaczach obowiązuje numeracja z pierwszej części). Będą również dyskutowane nowe przykłady programów i zadań. W ich prezentacji zastosowano tę samą notację, a numery im przyporządkowane stanowią kontynuację numeracji z pierwszej części artykułu.

Wykrywalność błędów przy testowaniu niesystematycznym, strukturalnym i funkcyjnym

Definicja: *Przypadek testowy będzie nazywany przypadkiem wykrywającym błąd, jeśli wynik otrzymany po uruchomieniu testowanego programu dla tego przypadku testowego będzie niepoprawny, czyli inny od wyniku wynikającego z zadania.*

Przykładowo, przypadek testowy (a, b) dla programu 1-1 jest przypadkiem wykrywającym błąd, jeśli $MAX = a$ i $a < b$ (wynik wynikający z zadania: maksimum jest równe b) lub jeśli $MAX = b$ i $a > b$ (wynik wynikający z zadania: maksimum jest równe a).

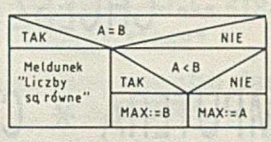
Zalety testowania strukturalnego w stosunku do testowania niesystematycznego i testowania funkcyjnego

Przy testowaniu strukturalnym każda część (tzn. przynajmniej każda gałązka) danego programu zostaje co najmniej raz wypróbowana. Fakt ten powoduje często optymistyczną ocenę zdolności testowania strukturalnego do wykrywania błędów w programie. Po bliższej analizie strukturalnych metod testowania okazuje się jednak, że te optymistyczne oceny są przesadzone – metody strukturalne nie zapewniają wykrycia wielu błędów, przy czym poszczególne metody testowania strukturalnego (np. testowanie gałązkowe i ścieżkowe) różnią się znacznie pod tym względem. Jednak fakt, że przy testowaniu strukturalnym systematycznie wykonuje się każdą część napisanego programu, prowadzi do istotnych korzyści w wykrywaniu błędów.

Rozważmy następujący przykład błędu: w gałązce *TAK* programu poprawnego dla zadania 1 zamiast $MAX := B$ napisano $MAX := A$. błąd ten zostałby na pewno wykryty przy testowaniu strukturalnym niepoprawnego programu (przez przypadek testowy dla gałązki *TAK*, spełniający warunek $a < b$, dla którego otrzymany wynik $MAX = a$ byłby w sposób oczywisty niepoprawny), przy testowaniu niesystematycznym – niekoniecznie.

Powyższy przykład wskazuje przewagę testowania strukturalnego nad testowaniem niesystematycznym (w testowaniu funkcyjnym błąd ten by również wykryto, por. [3] wybór 1-1.1 (f)). Następny przykład błędu pokaże, że testowanie strukturalne jest dla pewnych typów błędów lepsze także od testowania funkcyjnego.

Zadanie 1 nie wyróżnia w żaden sposób przypadku, w którym obie wczytywane liczby a i b są równe. Program realizujący to zadanie również nie musiałby zajmować się w sposób szczególny tym przypadkiem, np. w programie poprawnym dla zadania 1 (rys. 2 a w [3]) przypadek ten jest połączony z relacją $a > b$ w gałązce *NIE;* jednak programem poprawnym dla rozpatrywanego zadania jest również następujący program, który przy równych danych wejściowych a i b daje odpowiedni komunikat.



Rys. 1. Inny program poprawny dla zadania

Załóżmy, że programista popełnił przy pisaniu powyższego programu jakiś błąd w gałązce odpowiadającej przypadkowi $a = b$. Błąd ten przy testowaniu strukturalnym będzie na pewno wykryty, ponieważ testowanie strukturalne będzie wymagało uruchomienia programu właśnie z przypadkiem testowym (a, b) spełniającym warunek $a = b$, odpowiadającym gałązce zawierającej błąd.

Przy testowaniu funkcyjnym (a tym bardziej niesystematycznym) błąd powyższy nie musi zostać znaleziony, ponieważ sformułowanie zadania nie prowadzi bezpośrednio do uwzględnienia przypadku testowego $a = b$, np. wybór 1-1.1 (f).

W praktyce programy często zawierają gałązki, których istnienie nie wynika bezpośrednio ze sformułowania zadania, nawet przy uwzględnieniu innych charakterystyk zadania, jak np. końców przedziałów. Składają się na to następujące przyczyny:

- dokładne sformułowanie zadań na piśmie jest niepraktyczne; programista stosuje przy programowaniu dodatkowo swoją wiedzę i informacje ustne;

- wiele warunków i czynności wyrażonych w sformułowaniu zadania w prostej formie werbalnej wymaga przy realizacji programu bardziej złożonych struktur programowych.

Często stosuje się zasadę, że osoba testująca program jest niezależna od programisty, czyli dysponuje przy przeprowadzaniu testowania funkcyjnego tylko informacjami zawartymi bezpośrednio w sformułowaniu zadania. Wtedy testowanie funkcyjne nie może zapewnić wykrycia błędów popełnionych w poszczególnych gałązkach sprawdzanego programu. Nawet jeśli testujący, będący zwykle specjalistą w dziedzinie zastosowania programu, stara się generować dodatkowe funkcyjne przypadki testowe na podstawie swojej wiedzy, nie może on w ten sposób zapewnić wykrycia błędów w gałązkach powstałych z drugiej wymienionej przyczyny, ponieważ zależą one od zastosowanego przez programistę algorytmu, sposobu programowania i nie zawsze dadzą się wyprowadzić z samego zadania lub dziedziny przyszłego zastosowania programu. To wszystko powoduje, że testowanie strukturalne jest bardzo ważne dla weryfikacji programów komputerowych.

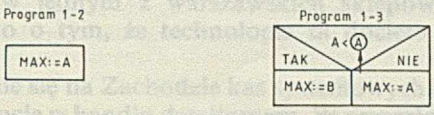
Testowanie strukturalne nie jest jednak wystarczające do zapewnienia wysokiego stopnia poprawności programów; konieczne jest równoległe stosowanie testowania funkcyjnego. Jego zalety wynikają głównie z faktu, że opiera się ono na sformułowaniu zadania, które przyjmuje się za poprawne, a nie – jak testowanie strukturalne – na programie, który być może zawiera błędy.

Zalety stosowania funkcyjnego w stosunku do testowania niesystematycznego i testowania strukturalnego

Wykrycie pewnych typów błędów nie może zostać w żaden sposób zagwarantowane przy testowaniu strukturalnym, a tym bardziej niesystematycznym. Chodzi tu o błędy dotyczące części programu zapomnianych przy programowaniu lub części niedostępnych.

Rozpatrzmy dwa następujące błędy, które programista mógłby popełnić przy pisaniu programu dla zadania 1.

W pierwszym z programów został zapomniany warunek; w programie drugim struktura programu pozornie nie została zmieniona, ale w wyniku błędu ścieżka *TAK* nie może zostać zrealizowana. Oba programy prowadzą do fałszywych wyników dla danych wejściowych a i b spełniających warunek $a < b$, natomiast dla $a >= b$ otrzymane wyniki są poprawne.



Rys. 2. Przykłady błędnych programów dla zadania 1

Każdy z powyższych błędów może być wykryty tylko wówczas, gdy zostanie wybrany przypadek testowy (a, b) spełniający warunek $a < b$. Tego nie można zagwarantować przy testowaniu strukturalnym: warunki na przypadki testowe dla gałązek przyjmują tu postać *true*; w obu przypadkach zostanie wybrany jeden przypadek testowy (a, b), który może spełniać zarówno warunek $a < b$, jak i $a > b$. Przy testowaniu funkcyjnym natomiast oba błędy z pewnością zostaną wykryte (por. [3], wybór 1-1.1 (f).

Istotne polepszenie wykrywalności błędów przy zastosowaniu kombinacji systematycznych metod testowania w stosunku do testowania niesystematycznego widać na zaprezentowanych przykładach. Niestety, kombinacja ta jest wciąż jeszcze niewystarczająca do zapewnienia wysokiej poprawności programów.

Ograniczenia zdolności kombinacji testowania strukturalnego i funkcyjnego do wykrywania błędów w programach komputerowych

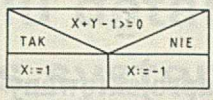
Omówimy teraz przykłady błędów w programach, które nie muszą zostać wykryte nawet po zastosowaniu zarówno testowania strukturalnego jak i testowania funkcyjnego. Własność tę mają np. błędy w programach 1-1 i 2-1.

W dalszym ciągu będą podane inne przykłady błędów w równie prostych programach (tzn. z dwoma gałązkami), w których prawdopodobieństwo niewykrycia błędu przy jednoczesnym zastosowaniu testowania strukturalnego i funkcyjnego będzie jeszcze większe. Przedtem jednak zostanie omówiona wykrywalność błędu z programu 2-1. Można sprawdzić, że program ten daje niepoprawne wyniki wtedy i tylko wtedy, gdy dane wejściowe a, b, c spełniają relację $b < c < a$. Do wykrycia tego błędu jest więc konieczny przypadek testowy (a, b, c) spełniający ten właśnie warunek. Przy testowaniu strukturalnym, a także przy testowaniu ścieżkowym, wybór takiego przypadku testowego nie jest zagwarantowany: warunek ten może być spełniony tylko przez przypadek 1-2.3 (s) IV, ale nie musi ($a > b$ i $b < c$ jest równoważne alternatywie $b < c < a$ lub $b < a < c$). Testowanie funkcyjne również nie zapewnia wykrycia rozpatrywanego błędu. I tak przy wyborze 1-2.1 (f) tylko przypadek 1-2. (f) I może spełnić warunek $b < c < a$, ale może on również spełniać warunek $c < b < a$ – wtedy błąd nie zostanie wykryty. Znalezienie rozpatrywanego błędu będzie zapewnione tylko wtedy, kiedy testowanie funkcyjne zostanie zrealizowane za pomocą wyboru 1-2.2 (f) (przypadek IV), a więc zależy ono od interpretacji *wszystkich możliwości wynikających z zadania.*

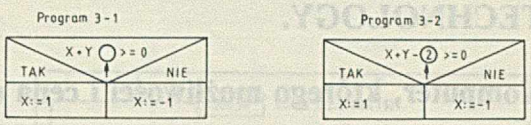
Rozpatrzmy teraz kolejny przykład zadania, przy którym łatwo popełnić inne, nie rozpatrywane do tej pory typy błędów.

Zadanie 3

Dla dowolnych liczb rzeczywistych x, y znaleźć znak liczby $x + y - 1$ (1 jeśli $x + y - 1$ jest nieujemne, -1 w przeciwnym razie).



Rys. 3. Przykład poprawnego programu dla zadania 3



Rys. 4. Przykłady błędów możliwych do popełnienia przy programowaniu zadania 3

Testowanie strukturalne oznacza dla programów 3-1 i 3-2 wybór dwóch przypadków testowych. Dla programu 3-1 pierwszy jest parą liczb (x, y) spełniających warunek $x + y \geqslant 0$, a drugi (x, y) spełniającą warunek $x + y < 0$. Przypadek drugi na pewno nie wykaże faktu, że program 3-1 zawiera błąd (dla liczb x, y spełniających warunek $x + y < 0$ wynik $x = -1$ jest poprawny według zadania 3), a przypadek pierwszy zrobi to tylko wówczas, gdy będzie dodatkowo spełniony warunek $x + y - 1 < 0$. Widać więc, że błąd popełniony w programie 3-1 zostanie wykryty przy testowaniu strukturalnym (gałązkowym lub ścieżkowym) z niewielkim prawdopodobieństwem. Sytuacja jest podobna w przypadku programu 3-2.

Również testowanie funkcyjne nie zapewnia wysokiego prawdopodobieństwa wykrycia błędów w programie 3-1 i 3-2. Jeżeli charakterystykę *wszystkie możliwości wynikające z zadania* zinterpretuje się jako zbiór dwóch możliwości: „wyrażenie $x + y - 1$ jest nieujemne" i „wyrażenie $x + y - 1$ jest ujemne", to zostaną wybrane dwa funkcyjne przypadki testowe, które wykryją rozpatrywane błędy z prawdopodobieństwem porównywalnym do prawdopodobieństwa osiągniętego przy testowaniu strukturalnym.

Łatwo sprawdzić, że pewne modyfikacje omówionych dotychczas metod testowania, polegające na wyborze dodatkowych, granicznych przypadków testowych, istotnie zwiększają prawdopodobieństwo (nawet do wartości bliskiej 1) wykrycia błędów rozpatrywanego typu. Graniczne przypadki testowe to przypadki z pogranicza warunków zadania lub testowanego programu; w zadaniu 3 i w programach 3-1 i 3-2 chodzi tu o wybór przypadków testowych spełniających odpowiednio warunki: $x + y = 0$ i $x + y - 2 = -\varepsilon$, gdzie ε jest bardzo małą liczbą dodatnią (problematykę tę dokładnie rozpatruje się w [1, 2, 5].

Niestety, również wybór przypadków granicznych nie zapewnia wykrycia stosunkowo łatwych do popełnienia błędów. Na przykład, błąd z programu 1-1 nie zostanie wykryty również po uwzględnieniu następujących przypadków testowych z pogranicza warunku $a < b$ (zadanie 1) i warunku $a < 8$ (program 1-1): $(a = 100, b = 100)$, $(a = 8, b = 3)$, $(a = 7.9, b = 100)$, $(a = 8.1, b = 2)$. Sytuacje takie powstają przy błędach polegających np. na pomyleniu operatora „+" z operatorem „−" lub pomyleniu nazw dwu zmiennych, np. X i Y (znaki te są umieszczone w niektórych klawiaturach obok siebie).

Istnieją wprawdzie dalsze możliwości modyfikacji rozpatrywanych metod testowania (omówione w cytowanej literaturze), które nakładają takie warunki na graniczne punkty testowe, że wykrycie kolejnej grupy błędów, także błędu z programu 1-1, jest zagwarantowane. Dotyczy to jednak tylko zadań i programów, w których występują jedynie operacje liniowe. Jeśli wykonuje się nieco bardziej skomplikowane obliczenia, jak mnożenie lub potęgowanie zmiennych, to różnorodność łatwych do popełnienia błędów staje się tak duża, że najdalej idące modyfikacje testowania nie mogą zapewnić ich wykrycia (przykłady znajdują się w [1]).

* * *

Testowanie jest najpopularniejszą formą weryfikacji programów komputerowych.

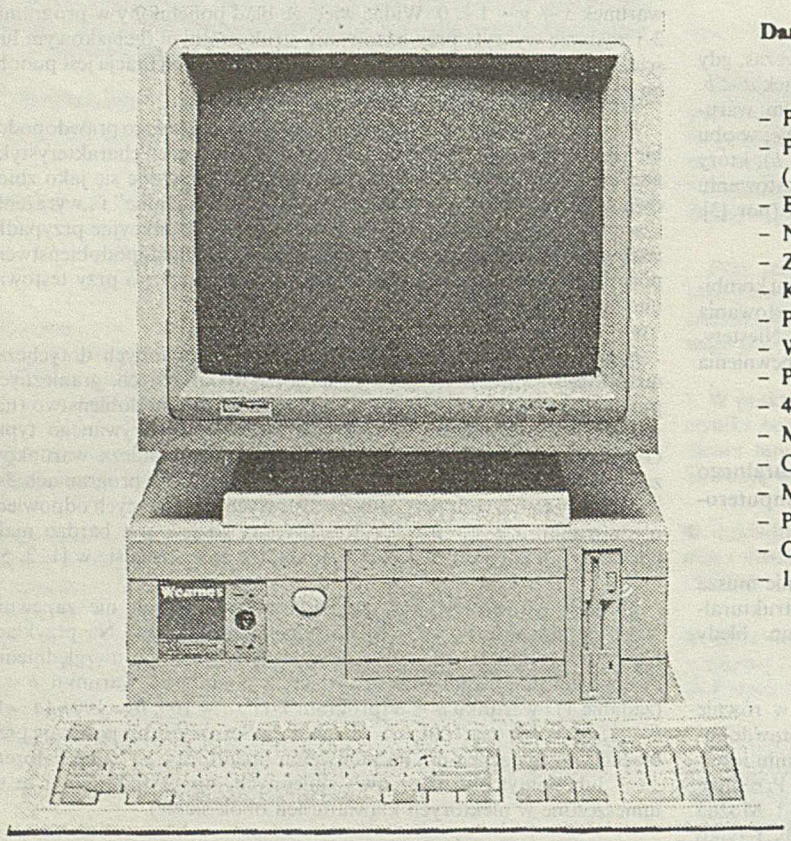
dokończenie na s. 28

KOMPUTER
Z WBUDOWANĄ PRZYSZŁOŚCIĄ

Komputer zaprojektowany przez czołowego światowego producenta mikrokomputerów – firmę ADVANCED LOGIC RESEARCH i wyprodukowany przez firmę WEARNES TECHNOLOGY.

Komputer, którego możliwości i cena oszałamiają konkurencję.

Komputer, który nie zestarzeje się, dzięki możliwości rozbudowy do architektury 386SX i 486.



Dane techniczne:

SERIA WEARNES BOLDLINE „M"

- Procesor 80286 – 12.5 Mhz
- Pamięć 1 MB RAM, możliwość rozbudowy do 16 MB (5 MB na płycie głównej)
- BIOS Phoenix
- Napęd dyskietek 3,5" 1,44 MB
- Zasilacz 110 Watt
- Klawiatura 101 klawiszy
- Podstawka dla koprocesora matematycznego 80287
- Wbudowany sterownik dyskowy z przeplotem 1:1
- Port szeregowy i równoległy
- 40 MB dysk sztywny
- Możliwość korzystania z EMS 4.0
- Obudowa typu „compact"
- Miejsce na dwa napędy 5,25" o wysokości 1/2
- Podręcznik i dyskietka z programem konfiguracyjnym
- Opcjonalna rozbudowa do 386SX i 486
- 12-miesięczna gwarancja

BOLDLINE COMPUTERS
GRUPA MICOMP-TECH
Biuro Informacji Techniczno-Handlowej
ul. Astrów 7, 40–045 Katowice
telefon i telefaks: 518-628
teleks: 315687 COMP PL

DYSTRYBUTORZY:
PTH „TECHMEX"
43–300 Bielsko-Biała
ul. M. Curie Skłodowskiej 13
tel.: 42-198, 47-555, telefaks: 47-624,
teleks: 35325

O/22/90

ANDRZEJ GÓRECKI
Melbourne, Australia

Komputeryzacja handlu detalicznego przy użyciu kas systemowych

Podróżujący po krajach rozwiniętych z pewnością są już przyzwyczajeni do elektronicznych (systemowych) kas sklepowych, z reguły wyposażonych w laserowe czytniki kodów kreskowych (ang. *bar codes*). Podczas mojej wizyty w Polsce (październik 1990) zauważyłem jednak, że sprzęt taki używano już w jednym z warszawskich sklepów Pewexu. Świadczy to o tym, że technologia ta dociera także do Polski.

Pojawienie się na Zachodzie kas systemowych spowodowało rewolucję w handlu detalicznym. W przeszłości polscy informatycy i handlowcy mieli ograniczone możliwości dostępu do tej technologii. Przyczyniła się do tego szczupłość literatury fachowej na ten temat. Istnieje mnóstwo publikacji na temat systemu księgowości informatycznej, płac, gospodarki materiałowej itp. Próby znalezienia podobnych źródeł na temat zastosowań komputerów w handlu detalicznym są nadal skazane na niepowodzenie. Korzyści płynące z użycia kas systemowych i ich integracji z centralnym systemem informatycznym przedsiębiorstwa dają tak dużą przewagę nad konkurencją, że nikt nie jest skłonny dzielić się w tej dziedzinie swoją wiedzą i doświadczeniem.

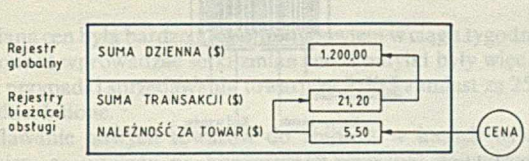
W artykule niniejszym chciałbym przełamać niepisane embargo na ten temat, przez omówienie podstawowych wiadomości o zastosowaniu informatyki w handlu detalicznym przy użyciu kas systemowych. Rozważę także zależności ekonomiczne, jakie należy brać pod uwagę przy ocenie celowości komputeryzacji sklepów.

Kasy systemowe

Wynalezione przed prawie stu laty i nadal szeroko stosowane w Polsce kasy mechaniczne spełniają dwie podstawowe funkcje:

- zliczają należności za zakupione przez klienta towary,
- dodają wszystkie kolejne przychody, aby na koniec dnia podać łączną sumę zebranych w kasie pieniędzy od chwili otwarcia sklepu.

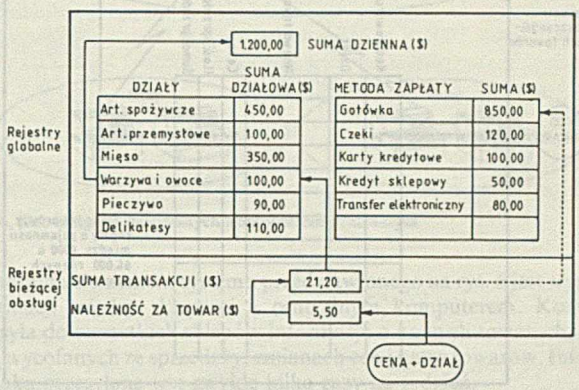
Podstawowe elementy logiczne takiej kasy przedstawia rysunek 1.



Rys. 1. Podstawowe rejestry kasy mechanicznej

Rejestry Bieżącej Obsługi zliczają należności od klientów. Cena każdego sprzedanego towaru jest wprowadzana do rejestru NALEŻNOŚĆ ZA TOWAR, skąd poszczególne należności jednego klienta są przenoszone i dodawane w rejestrze SUMA TRANSAKCJI. Z chwilą zakończenia transakcji z klientem rejest SUMA DZIENNA zostaje powiększony o pobraną od klienta należność. Na koniec dnia rejestr ten jest ustawiany na zero.

Następny szczebel rozwoju stanowiły kasy elektryczne, które przyniosły wiele usprawnień, a przy okazji całą procedurę znacznie skomplikowały. Podstawowe elementy logiczne takiej kasy zilustrowano na rysunku 2.



Rys. 2. Podstawowe rejestry kasy elektrycznej

Kasjer wybija na klawiaturze należność za każdy towar, a następnie naciska klawisz identyfikujący dział. Należność za towar jest dodawana równocześnie do zawartości trzech rejestrów:

- SUMA TRANSAKCJI (z klientem),
- SUMA DZIAŁOWA,
- SUMA DZIENNA.

Na zakończenie transakcji z klientem następuje dodatkowo uaktualnienie jednego z liczników rejestrujących metodę zapłaty. Na koniec dnia sumy wszystkich działów muszą równać się sumie dziennej oraz sumie pieniędzy pobranych od klientów we wszystkich postaciach (gotówka, czeki itp.).

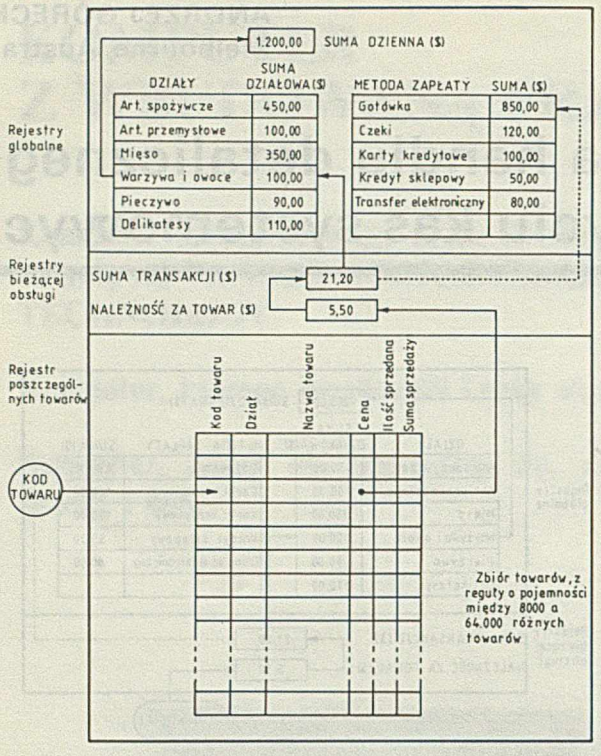
W przedstawionym przykładzie kasa pobrała od początku dnia (tj. od momentu, gdy wszystkie rejestry zostały ustawione na zero) 350,00 $ za towary z działu mięsnego, 1.200,00 $ za towary ze wszystkich działów, itp. Stan rejestrów kasy przedstawiono w chwili, gdy obsługiwano klienta. Właśnie wybito na kasie należność 5,50 $, a całkowita należność klienta za towary już przetworzone przez kasę wynosi 21,20 $. Po pobraniu końcowej należności rejestry zawierające 5,50 $ oraz 21,20 $ są ustawiane na zero i oczekują na następnego klienta.

Kasy elektryczne pozwalają śledzić sprzedaż w poszczególnych działach, co w powiązaniu z rejestracją dostaw na poziomie działu daje możliwość śledzenia przepływu towarów przez sklep i przybliżony pomiar zysku lub strat. Obliczenia takie opierają się na założeniu, że marża detaliczna jest w poszczególnych działach identyczna.

W sklepie wyposażonym w kasy elektryczne każdy wybrany przez klienta towar nadal musi mieć albo cenę zaznaczoną na opakowaniu, albo kasjer musi pamiętać ceny wszystkich towarów. Obie te metody są podatne na błędy, zwłaszcza w sytuacji zmiany cen.

Następny szczebel rozwoju kasy sklepowej to kasa systemowa. Jednym z pionierów w tej dziedzinie była firma DTS (*Data Terminal Systems*), która rozpoczęła masową produkcję tych urządzeń. Do chwili obecnej na świecie zainstalowano już dziesiątki tysięcy kas systemowych. Urządzenia tego typu dostarczają głównie firmy NCR, TEC, IBM, Nixdorf itd.

Podstawowe elementy logiczne kasy systemowej zilustrowano na rys. 3.



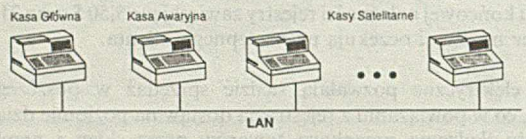
Rys. 3. Podstawowe rejestry kasy systemowej

Obsługa klienta polega na wybijaniu na klawiaturze lub czytaniu za pomocą czytnika laserowego kodu towaru. Kod ten pozwala na wyszukanie w pamięci kasy ceny towaru oraz jego działu. Każdy towar (lub jego wielokrotność) jest rejestrowany w dwóch polach rekordu towaru: liczniku ilości sprzedanego towaru oraz w sumatorze wpływów ze sprzedaży tego towaru. Po zakończeniu tej operacji następuje uaktualnienie rejestrów: sumy sprzedaży danego działu, sumy sprzedaży dziennej i sumy należności od klienta. Po otrzymaniu należności od klienta następuje uaktualnienie jednego z rejestrów sposobu zapłaty.

Kasy systemowe pozwalają śledzić sprzedaż poszczególnych towarów. Ceny nie muszą być już umieszczane na opakowaniach, bowiem kod towaru pozwala na wyszukanie ceny w pamięci kasy systemowej. Zmiany cen towarów mogą odbywać się więc błyskawicznie, wystarczy tylko zmienić cenę w pamięci kasy.

System sklepowy

Kasy systemowe są z reguły używane w zestawach złożonych z kilku do kilkunastu kas. Schemat systemu sklepowego pokazano na rys. 4.



Rys. 4. Typowa konfiguracja kas systemowych

Przedstawiony system jest wyposażony w dwa identyczne zbiory zawierające pełną listę dostępnych w sklepie towarów. Każdy zbiór umieszczono w pamięci oddzielnego komputera, co oznacza, że system kasowy ma dwa wyspecjalizowane komputery. Na rysunku 4 pokazane są one jako Kasa Główna (ang. *Master Register*) oraz Kasa Awaryjna (ang. *Backup Register*).

Użycie dwóch identycznych zbiorów ma zadanie redukcji ryzyka utraty informacji oraz zapewnia ciągłość obsługi klientów w przypadku awarii jednego z komputerów. Jak pokazano na rysunku, do systemu można dołączyć Kasy Satelitarne (ang. *Slaves* lub *Satellites*), które są sterowane przez oba komputery. Kasy takie są w zasadzie wyspecjalizowani terminalami i podczas obsługi klienta wprowadzony do Kasy Satelitarnej kod towaru jest przesyłany do Kasy Głównej, która odczytuje cenę, dział i nazwę towaru ze swojego zbioru i przesyła te informacje z powrotem do Kasy Satelitarnej.

Kasy komunikują się z komputerami (tzn. Kasą Główną i Awaryjną) za pomocą sieci lokalnej LAN (ang. *Local Area Network*). W większości stosowanych systemów kasowych protokoły komunikacyjne LAN są niestandardowe, co oznacza, że włączenie dodatkowych urządzeń do systemu kasowego może być z reguły dokonane tylko przez producenta sprzętu, znającego szczegóły protokołu.

Kasy systemowe pozwalają także realizować koncepcję „ruchomego kasjera" (ang. *floating cashier*). Ruchomy kasjer polega na tym, że system rejestruje informacje o kasjerach i dla każdego z nich przechowuje dane o zebranych środkach płatniczych oraz początkowej zawartości kasy. Pozwala to w każdej chwili obliczyć sumę pieniędzy jaką kasjer powinien mieć w szufladzie kasy. Warto zauważyć, że system kontroluje kasjerów, a nie kasy. Każdy kasjer ma swoją szufladę z pieniędzmi, którą wkłada do kasy zaczynając pracę. Powoduje to równocześnie zarejestrowanie w systemie obecności tego kasjera. Po zakończeniu pracy kasjer powiadamia system o zakończeniu pracy i zabiera ze sobą szufladę. Szuflady są wyposażone w pokrywę i ze względów bezpieczeństwa zamykane na klucz. Zapobiega to ewentualnej kradzieży lub zgubie zawartości.

Kody kreskowe

Kasy systemowe w dużej mierze zawdzięczają swoje powodzenie kodom kreskowym. Jak już wspomniano, kody te można odczytywać za pomocą czytników laserowych, co pozwala na błyskawiczną obsługę klientów. Kody kreskowe stają się tak popularne, że coraz trudniej jest znaleźć towar nie mający takiego kodu.

Dzięki kodom kreskowym proces sprzedaży za pomocą kas systemowych wygląda zupełnie inaczej niż odbywało się to przy użyciu tradycyjnego sprzętu. Podstawowe różnice są następujące:

- Ceny towarów nie umieszcza się już na opakowaniu (przez producenta lub przez personel sklepu). Występują one na półce w postaci etykietki, która jest zmieniana przy każdorazowej zmianie ceny towaru.
- Kasjer przesuwa wybrane przez klienta towary nad umieszczonym w stoliku kasy czytnikiem laserowym lub używa czytnika ręcznego (ang. *laser gun*), co powoduje wprowadzenie kodu kreskowego do pamięci kasy. Dla każdego prawidłowego kodu kasa wyszukuje zapisaną w pamięci cenę towaru.
- Towary, które na opakowaniu nie mają kodów kreskowych, są znakowane w sklepie tzw. kodami szybkimi (1 do 4 cyfr). Kody takie (cyfrowe) oczywiście nie mogą być rozpoznane przez czytnik i kasjerzy muszą wprowadzać je ręcznie, używając klawiatury kasy.

Kody kreskowe występują w kilku odmianach. Najczęściej spotykany kod kreskowy EAN (European Article Number) wygląda następująco (rys. 5):

Rys. 5. Przykład kodu kreskowego EAN

Światło wysyłane z czytnika przebiegając przez kreski kodu jest modulowane a jego odbicie jest analizowane przez układy logiczne czytnika. Przeanalizowany sygnał jest zamieniany na ciąg cyfr i przesyłany do kasy. Każda cyfra jest reprezentowana przez różnej grubości kreski i zmiennej szerokości odstępy między kreskami.

W Stanach Zjednoczonych są używane kody UPC (*Universal Product Code*), będące podzbiorem EAN. Kody te zawierają tylko 11 cyfr i kasy interpretują je jako EAN = 00 + UPC.

Kody EAN oraz UPC występują także w postaci skróconej. Są stosowane na tych wyrobach, które są zbyt małe, aby na ich opakowaniu można było wydrukować pełny kod.

Specjalne znaczenie mają kody EAN rozpoczynające się cyfrą. Kody kraju od 20 do 29 nie zostały bowiem przyznane żadnemu państwu i są zarezerwowane do użytku wewnątrzsklepowego. Oznacza to, że sklep może wydrukować własne kody kreskowe bez obawy, że jakikolwiek inny towar będzie miał identyczny kod. Dostawca towaru bowiem musi używać kodu kraju.

Kody kreskowe są z reguły drukowane w sklepie (używając specjalnej drukarki lub elektronicznej wagi sprzężonej z drukarką) z następujących powodów:

- Sklep może przyjąć zasadę, że wszystkie towary w sklepie muszą mieć kod kreskowy. W takiej sytuacji dla tych towarów, które nie mają kodów kreskowych umieszczonych na opakowaniu przez producenta drukuje się samoprzylepne etykietki z kodami kreskowymi. Dostawca towaru nieoznakowanego jest z reguły obciążany opłatą do 5 centów za każdą etykietkę.

- Towary sprzedawane na wagę oczywiście nie mają kodów kreskowych. Towary takie pochodzą zwykle z działów mięsnego, delikatesowego i owocowo-warzywnego. Dla znakowania towarów odważanych używa się wag elektronicznych, które automatycznie drukują etykietkę z opisem towaru i kodem kreskowym.

Kody kreskowe drukowane w sklepach mają inną strukturę niż typowe kody EAN. Stosowane są dwa rodzaje kodów: zawierające należność lub wagę towaru. Kasa systemowa, która dysponuje informacją o cenie za 1 kg przelicza ciężar towaru na należność i odwrotnie. Informacja o ciężarze towaru jest konieczna dla uaktualnienia rejestru ilości sprzedanego towaru, natomiast należność jest dodawana do sumy pieniędzy uzyskanych ze sprzedaży towaru od czasu ostatniego wyzerowania kasy.

Wagę systemową drukującą etykietki z kodami kreskowymi można nabyć w zależności od wersji i producenta za ok. 3000 do 7000 $. Wagi takie mogą być programowane centralnie, używając komputera. W praktyce jednak wiele przedsiębiorstw nadal wprowadza i koryguje dane ręcznie.

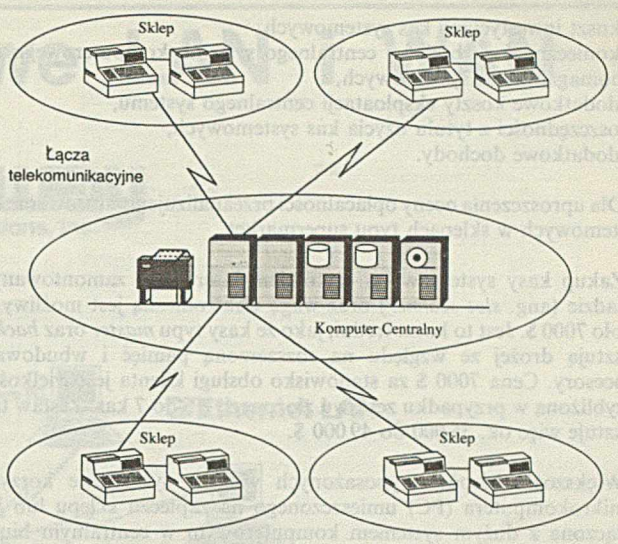
Centralny system komputerowy

Jak już wspomniałem, kasy systemowe diametralnie zmieniają sposób działania przedsiębiorstwa handlowego. Działalność sklepów może być śledzona na poziomie poszczególnych towarów, a nie działów, jak to odbywało się dotychczas. Możliwe staje się precyzyjne przewidywanie popytu na podstawie statystyki sprzedaży, sterowania zapasami w sklepach itp. Nie bez znaczenia jest także przyspieszenie obsługi w kasach i eliminacja konieczności tradycyjnego znakowania cen na każdej jednostce towaru.

Wydawać by się więc mogło, że kasy systemowe powinny rozpowszechnić się błyskawicznie. Wbrew jednak pozornej logice pierwsi użytkownicy kas systemowych szybko zniechęcili się do tej nowej techniki. Przyczyny były różnorakie:

- Zmiana cen była bardzo kłopotliwa, bowiem w ciągu tygodnia każdy sklep musiał wprowadzać setki zmian cen. Pomyłki były więc nieuniknione i przypadki sprzedawania towaru za 2,50 $ zamiast za 25,00 $ nie były odosobnione.
- Dodawanie nowych towarów do zbiorów w kasach było jeszcze trudniejsze ze względu na konieczność używania do wprowadzania opisu towarów klawiatury numerycznej (na przykład: A = 01, B = 02 itp.).
- Wydruki (jakkolwiek z doskonałą informacją), dostępne tylko na wąskich rolkach kasowych, były trudne do odczytania.
- Informacja o sprzedaży miała ograniczoną użyteczność, gdyż zbiory w kasach nie zawierały informacji o koszcie nabycia towarów. Nie pozwalało to na (teoretycznie możliwą) pełną kontrolę zysku na poziomie poszczególnych towarów.

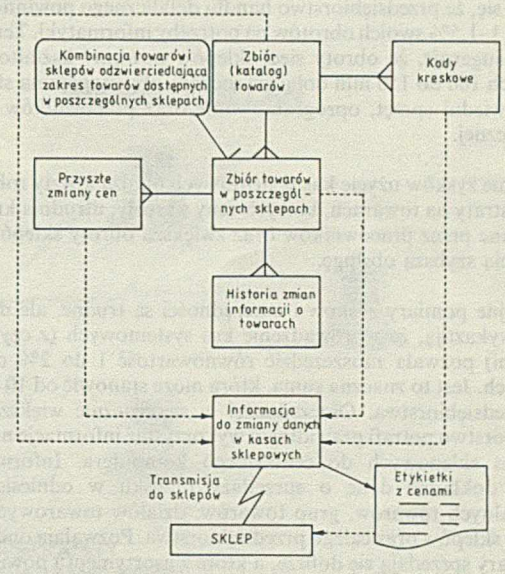
Bardzo szybko stało się oczywiste, że konieczne jest przyłączenie kas sklepowych do oddzielnego komputera dla uzyskania systemu, który byłby praktyczny w warunkach sklepowych. Rozwiązanie takie okazało się szczególnie skuteczne w odniesieniu do sieci sklepów. Schemat systemu dla sieci sklepów wyposażonych w kasy systemowe jest pokazany na rys. 6.



Rys. 6. Sieć sklepów sterowanych komputerem centralnym

Istotnym elementem systemu przedstawionego na rys. 6 jest wymiana informacji między sklepami a centralnym komputerem. Komputer przesyła do wszystkich sklepów informacje o nowych towarach, towarach wycofanych ze sprzedaży, zmianach cen i nazw towarów. Informacja taka przesyłana jest zwykle kilka razy na tydzień.

Systemy sklepowe przesyłają do centralnego komputera informacje o sprzedaży poszczególnych wyrobów. Transmisje takie odbywają się codziennie lub raz na tydzień. Dane ze sprzedaży porównywane są w komputerze z kosztem nabycia towarów, co pozwala na precyzyjne obliczanie zysków zakumulowanych od czasu ostatniej transmisji danych ze sklepów. Informacje te z reguły są składowane w pamięci dyskowej (nierzadko do dwu lat) i używane do prognozowania sprzedaży oraz oczekiwanych zysków. Rysunek 7 przedstawia podstawowe elementy danych w scentralizowanym systemie sterowania kasami systemowymi.



Rys. 7. Podstawowe elementy danych centralnego systemu sterowania siecią sklepów

Transmisja danych między sklepami a centralnym komputerem z reguły odbywa się za pomocą łączy komutowanych o przepustowości 2400 bitów/s. Niektóre sieci sklepów zaczynają obecnie używać łącz stałych, co jest jednak bardzo kosztowne.

Opłacalność komputeryzacji w sklepach

Rozważając opłacalność stosowania kas systemowych musimy wziąć pod uwagę następujące czynniki:

- koszt inwestycyjny kas systemowych,
- konieczność rozbudowy centralnego systemu komputerowego dla wspomagania kas systemowych,
- dodatkowe koszty eksploatacji centralnego systemu,
- oszczędności z tytułu użycia kas systemowych,
- dodatkowe dochody.

Dla uproszczenia oceny opłacalności przeanalizujemy stosowanie kas systemowych w sklepach typu supermarket.

Zakup kasy systemowej z czytnikiem laserowym zamontowanym w ladzie (ang. *slot scanner*) oraz wagą elektroniczną jest możliwy za około 7000 $. Jest to koszt średni, jako że kasy typu *master* oraz *backup* kosztują drożej ze względu na rozszerzoną pamięć i wbudowane procesory. Cena 7000 $ za stanowisko obsługi klienta jest wielkością przybliżoną w przypadku zestawu złożonego z 5 do 7 kas. Zestaw taki kosztuje więc ok. 35 000 do 49 000 $.

Większość sklepów wyposażonych w kasy systemowe korzysta z mikrokomputera (PC) umieszczonego na zapleczu sklepu lub jest połączona z dużym systemem komputerowym w centralnym biurze firmy. Wybór jednego z powyższych rozwiązań zależy od tego, czy sklep prowadzi działalność indywidualną czy też jest jednym ze sklepów sieci.

W tradycyjnym przedsiębiorstwie handlowym centralny komputer był używany głównie do księgowości informatycznej, rozliczeń finansowych oraz obliczania płac. Bardziej zaawansowane przedsiębiorstwa używały komputerów także dla sterowania systemem należności, który jest konieczny do kontroli przyznawanych rabatów od dostawców oraz do kontroli zakupu i sprzedaży towarów. Rozszerzenie zakresu zastosowań centralnego systemu komputerowego do sterowania kasami systemowymi z reguły wymaga rozbudowy systemu. Można założyć, że ok. 30 do 50% kosztu takiego rozbudowanego systemu jest związane z kasami systemowymi.

Użycie dużego systemu komputerowego, ze względu na jego koszt, jest opłacalne tylko w przypadku sieci składającej się z minimum 20 supermarketów. Kluczem jest tu wielkość obrotów. Koszt prowadzenia wydziału komputerowego średniej wielkości wynosi około 1,5 mln dolarów rocznie (jeżeli nie używa się dużych systemów komputerowych, które mogą kosztować nawet powyżej 4 mln $ rocznie). Jako regułę przyjmuje się, że przedsiębiorstwo handlu detalicznego powinno wydawać około 1–1,5% swoich obrotów na potrzeby informatyki. Ten prosty rachunek sugeruje, że obroty sieci sklepów powinny kształtować się w granicach 100 do 150 mln dolarów rocznie, aby mogła ona sfinansować odpowiedni sprzęt, oprogramowanie oraz pracowników obsługi informatycznej.

Po stronie zysków użycie kas systemowych obniża koszty robocizny, eliminuje straty na towarach, których ceny wzrosły, utrudnia kradzieże dokonywane przez pracowników oraz zwiększa obroty sklepów, gdyż klienci lubią szybszą obsługę.

Precyzyjne pomiary zysków i oszczędności są trudne, ale dostępne badania wykazują, że wprowadzenie kas systemowych (z czytnikami laserowymi) pozwala zaoszczędzić równowartość 1 do 2% obrotów towarowych. Jest to znaczna suma, która może stanowić od 10 do 20% zysku przedsiębiorstwa. Oszczędności te są znacznie większe, jeżeli przedsiębiorstwo potrafi prawidłowo wykorzystać informacje napływające z kas sklepowych do centralnego komputera. Informacje te obejmują dokładne dane o sprzedaży i zysku w odniesieniu do indywidualnych towarów, grup towarów, działów towarowych, sklepów, grup sklepów oraz całego przedsiębiorstwa. Pozwalają one ustalić, które towary sprzedają się dobrze, a które z asortymentu powinny być wyeliminowane. Kierownictwo sieci sklepów jest w stanie ustalić, które towary przynoszą największe zyski, aby umieścić je w najbardziej eksponowanych miejscach sklepu. Dostęp do dokładnej informacji o ilościach i wartości sprzedanych towarów dostarczanych przez danego producenta jest również doskonałym orężem w czasie negocjacji z dostawcami. Porównując koszty z oczekiwanymi korzyściami należy rozważyć następujące fakty:

1. Praktyka wykazuje, że pojedyncza kasa może obsłużyć sprzedaż wartości ok. 15 000 $ na tydzień. Oznacza to, że supermarket wyposażony w 6 kas może prawidłowo obsłużyć obroty towarowe rzędu 100 tys. $ tygodniowo, co stanowi około 5 mln dolarów w stosunku rocznym. Wprowadzenie kas systemowych pozwala zaoszczędzić 1,5% wartości obrotów, co daje roczną oszczędność w kwocie ok. 75 000 $.

2. Rozważając sieć 20 supermarketów wydającą 1,5 mln dolarów rocznie na centralny system komputerowy, koszt przypadający na każdy sklep wynosi ok. 75 000 $ rocznie. Zakładając, że 40% kosztu centralnego systemu jest związane z obsługą kas systemowych, na każdy sklep przypada ok. 30 000 $ rocznie. Zmniejsza to wysokość oszczędności do ok. 45 000 $ rocznie.

3. Biorąc pod uwagę koszt zakupu sześciu kas wynoszący ok. 42 tys. $ otrzymujemy okres amortyzacji takiej inwestycji poniżej 12 miesięcy.

4. Dodatkowe zyski, nie uwzględnione w powyższych rozważaniach:

- lepszy dobór asortymentów dzięki szczegółowym statystykom sprzedaży poszczególnych towarów,
- opracowanie wytycznych odnośnie sposobu umieszczania towarów na półkach sklepowych (towar umieszczony na wysokości oczu nabywcy sprzedaje się czterokrotnie lepiej niż towar umieszczony na podłodze sklepu),
- tańsze zakupy od dostawców (większa siła nabywcza wynikająca z dokładnej informacji na temat szybkości sprzedaży towarów poszczególnych dostawców),
- szybka identyfikacja problemów sprzedaży, pojawiających się w poszczególnych sklepach i działach,
- kontrola szybkości obrotu towarów (tzn. wskaźnik kosztu zapasów w porównaniu do dochodu ze sprzedaży).

Z powyższych faktów wynika, że użycie kas systemowych jest wysoce opłacalne.

Kasy systemowe w Polsce

Szerokie stosowanie kas systemowych na Zachodzie potwierdza, że są one opłacalne. Musimy jednak pamiętać, że potwierdza to jedynie fakt opłacalności tych kas... w krajach zachodnich. W warunkach polskich sytuacja jest odmienna, gdyż opłacalność użycia tej technologii jest uwarunkowana pełnym uzyskaniem wszystkich oczekiwanych oszczędności. W warunkach polskich nie jest to jeszcze możliwe, gdyż znaczna część oszczędności wynika z mniejszej pracochłonności, a więc kosztu robocizny związanego ze sprzedażą (obsługą klienta).

W typowym supermarkecie w krajach zachodnich koszt robocizny waha się od 5% do 12% sumy obrotów towarowych. Dopóki koszt robocizny w polskich supermarketach nie osiągnie minimum 6–7% sumy obrotów, oszczędności z tytułu zmniejszenia kosztu robocizny będą zbyt małe. Dodatkową barierą utrudniającą wprowadzenie kas z czytnikami laserowymi jest zupełny brak kodów kreskowych na polskich wyrobach. Ostatnim, choć nie mniej ważnym elementem, jest niedostateczny jeszcze rozwój konkurencyjnego rynku hurtowego, gdzie precyzyjne informacje o zachowaniu poszczególnych towarów mogą być użyte z korzyścią dla nabywcy, tj. przedsiębiorstwa handlu detalicznego.

Podsumowując uważam, że warunki w polskim handlu detalicznym nie są jeszcze odpowiednie dla szerokiego wprowadzania kas systemowych. Spodziewam się, że sytuacja ta powinna ulec zmianie w ciągu najbliższych 12–18 miesięcy.

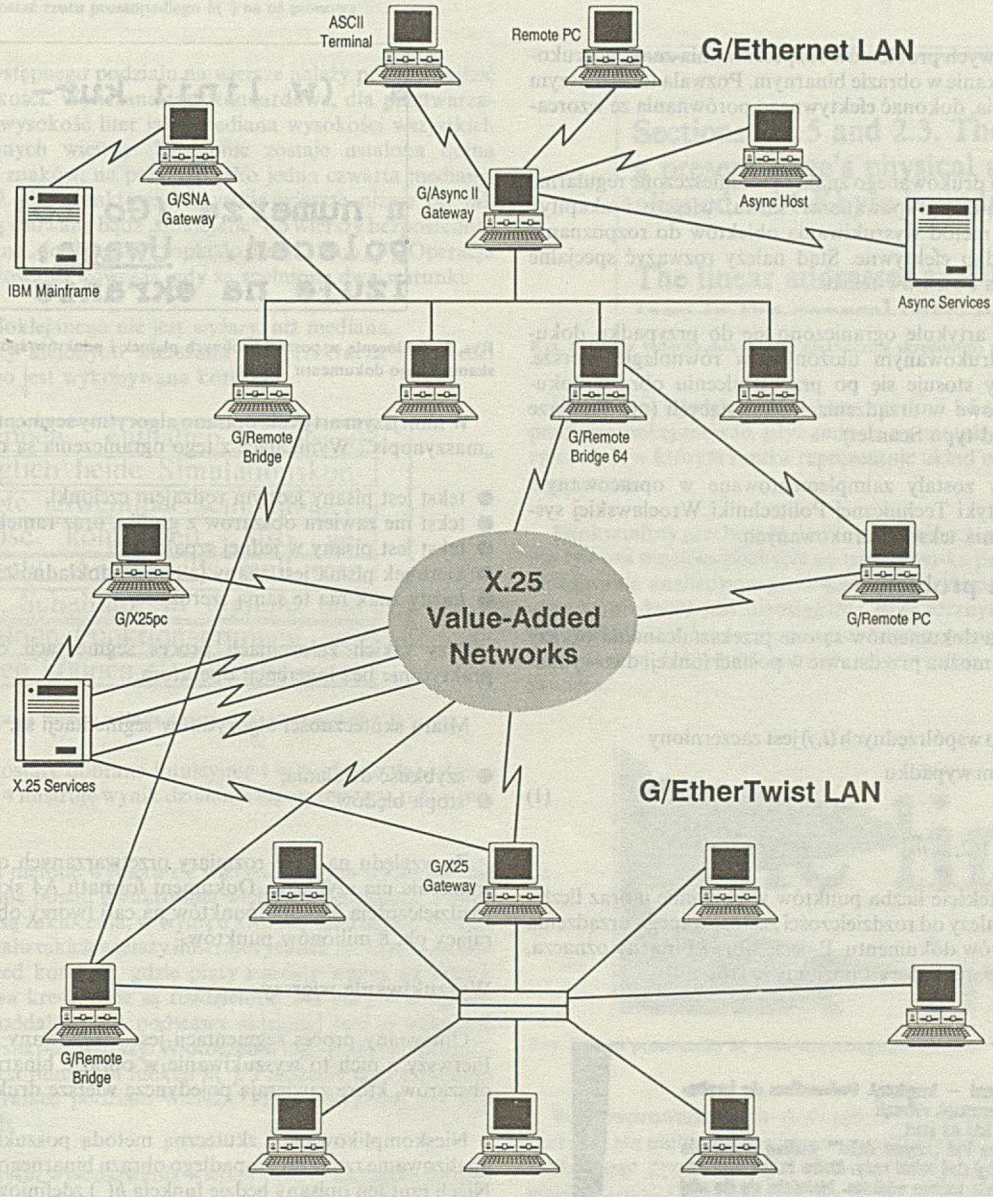
Handel polski ma tu szczególną szansę. W ciągu następnego roku można przygotować gospodarkę i społeczeństwo do wprowadzenia tej nowej technologii. Zamiast polegać wyłącznie na mechanizmach rynkowych, władze polskie mogą wykorzystać ten okres dla wprowadzenia przepisów nakazujących obowiązkowe oznaczanie wybranych kategorii wyrobów polskich za pomocą kodów kreskowych. Byłoby to znacznym ułatwieniem w momencie szerokiego wprowadzenia kas systemowych do polskiego handlu. Pomogłoby to również w pewnym stopniu w eksporcie polskich wyrobów do tych krajów, gdzie kody kreskowe są bezwzględnie wymagane.

Ważnym krokiem w tym kierunku byłoby powołanie organizacji PNW czyli Polskiej Numeracji Wyrobów. PNW powinna zrzeszać wszystkie organizacje, które używają kodów kreskowych, tzn. producentów wyrobów oraz przedsiębiorstwa handlu detalicznego.

dokończenie na s. 28

Produkty sieciowe LAN i WAN

Gateway
communications, inc.



BEZKONKURENCYJNE PRODUKTY
SIECI KOMPUTEROWYCH NAGRADZANE PRZEZ:
PC MAGAZINE, LAN MAGAZINE, INFO WORLD
OFERUJE AUTORYZOWANY DYSTRYBUTOR:

MIKROB

P.W.P.T. MIKROB SP. Z O.O.
20-601 LUBLIN, UL. ZANA 38 BLOK C, VIII p.
TEL. (0-81) 55-90-19, **FAX** 55-91-69, **TLX** 643776 MIKRO PL

Na życzenie wysyłamy katalog produktów. Atrakcyjny program współpracy dla dealer'ów.

CEZARY GRZEGORZ DOŁĘGA
Instytut Cybernetyki Technicznej
Politechnika Wrocławska

Problemy wyszukiwania znaków w rozpoznawaniu tekstu drukowanego

Jednym z podstawowych problemów rozpoznawania znaków drukowanych jest ich wyszukanie w obrazie binarnym. Pozwala to, w dalszym procesie rozpoznawania, dokonać efektywnego porównania ze wzorcami liter.

W przypadku tekstu drukowanego znaki są rozmieszczone regularnie w obszarze dokumentu (zgrupowane w słowa, wiersze i akapity). Stosowanie ogólnych metod wyszukiwania obiektów do rozpoznania nie jest w tym wypadku efektywne. Stąd należy rozważyć specjalne algorytmy wydzielania obszarów tekstu.

W prezentowanym artykule ograniczono się do przypadku dokumentów z tekstem drukowanym ułożonym w równoległe wiersze. Rozważane algorytmy stosuje się po przekształceniu obrazu dokumentów na mapy bitowe w urządzeniach skanujących (np. skanerze firmy Helwett Packard typu ScanJet).

Opisane algorytmy zostały zaimplementowane w opracowanym w Instytucie Cybernetyki Technicznej Politechniki Wrocławskiej systemie do rozpoznawania tekstów drukowanych.

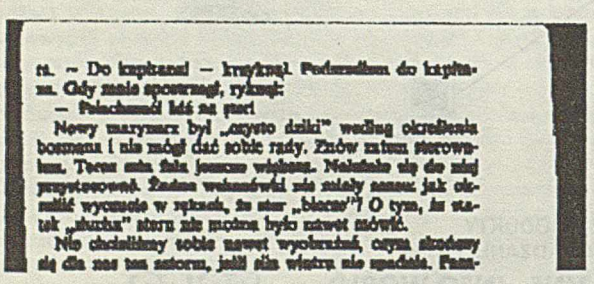
Charakterystyka problemu

W procesie czytania dokumentów są one przekształcane na obrazy binarne. Obrazy takie można przedstawić w postaci funkcji dwuwymiarowej $p(i, j)$:

$$p_{ij} = \begin{cases} 1 & \text{jeżeli punkt o współrzędnych } (i,j) \text{ jest zaczerniony} \\ 0 & \text{w przeciwnym wypadku} \end{cases} \quad (1)$$

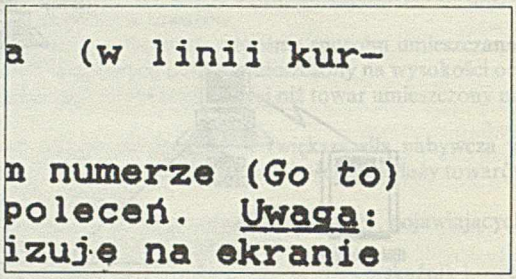
dla $i \in \{1, 2, ..., n\}$, $j \in \{1, 2, ..., m\}$

W zeskanowanym tekście liczba punktów w poziomie n oraz liczba punktów w pionie m zależy od rozdzielczości zastosowanego urządzenia skanującego i rozmiarów dokumentu. Pojęcie **obraz binarny** oznacza, że nie są odwzorowywane kolory i poziomy szarości.



Rys. 1. Zakłócenia w obrazie binarnym typu „czarne marginesy"

Wyszukiwanie tekstu w obrazie binarnym polega na szukaniu takich jego fragmentów, które zawierają znaki drukarskie (segmentacja obrazu). W ogólnym przypadku jest to zadanie bardzo skomplikowane. Różnorodność tekstów uniemożliwia praktycznie stworzenie systemu całkowicie niezawodnego, czyli rozpoznającego każdy ze spotykanych rodzajów dokumentów. Rysunki 1 i 2 ilustrują dwa spośród wielu zakłóceń, jakie spotkać można w obrazach binarnych.



Rys. 2. Zakłócenia w postaci drobnych plamek i punktów spowodowane złą jakością skanowanego dokumentu

W niniejszym artykule opisano algorytmy segmentacji dla druku typu „maszynopis". Wynikające z tego ograniczenia są następujące:

- tekst jest pisany jednym rodzajem czcionki,
- tekst nie zawiera obszarów z grafiką oraz ramek,
- tekst jest pisany w jednej szpalcie,
- kierunek pisma jest znany (z pewną dokładnością),
- każdy znak ma tę samą szerokość.

Przy takich założeniach proces segmentacji odbywać się może praktycznie bez ingerencji operatora.

Miarą skuteczności algorytmów segmentacji są:

- szybkość działania,
- stopa błędów.

Ze względu na duże rozmiary przetwarzanych obrazów szczególne znaczenie ma szybkość. Dokument formatu A4 skanowany z typową rozdzielczością 300 dpi (punktów na cal) tworzy obraz binarny zawierający ok. 8 milionów punktów.

Wyszukiwanie wierszy

Omawiany proces segmentacji jest realizowany w dwóch krokach. Pierwszy z nich to wyszukiwanie w obrazie binarnym $p(\ )$ tych jego obszarów, które zawierają pojedyncze wiersze druku.

Nieskomplikowaną i skuteczną metodą poszukiwania wierszy jest analizowanie rzutu prostopadłego obrazu binarnego na oś pionową [1]. Niech rzut ten opisany będzie funkcją $h(\ )$ zdefiniowaną w następujący sposób:

$$h(i) = \sum_{i=1}^{n} p(i,j) \quad (2)$$

dla $i \in \{1, 2, ..., m\}$

Przykładowy wynik rzutowania ilustruje rysunek 3.

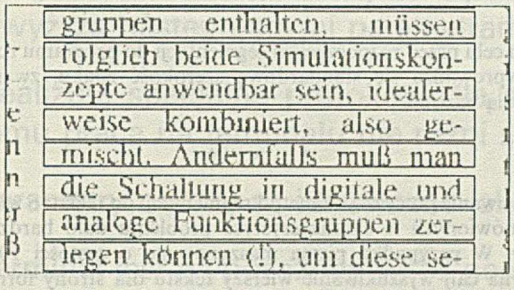
Granice wierszy są wyznaczane przez miejsca, gdzie rzut zmienia wartość z zera na dodatnią lub odwrotnie. Aby wyeliminować część zakłóceń w postaci pojedynczych punktów, w przerwach między wierszami, które wprowadzają fałszywe linie tekstu, należy podnieść próg dyskryminacji. W analizowanych przykładowych dokumentach

stwierdzono doświadczalnie, że najlepsze wyniki dawał próg dwóch punktów.

Rys. 3. Przykładowa postać rzutu prostopadłego $h(\ )$ na oś pionową

Po dokonaniu wstępnego podziału na wiersze należy przeprowadzić korekcję ich wysokości. Wyliczana jest standardowa, dla przetwarzanego dokumentu, wysokość liter jako mediana wysokości wszystkich wcześniej znalezionych wierszy. Następnie zostaje ustalona dolna granica wysokości znaków, na przykład jako jedna czwarta mediany, lecz nie mniej, niż pięć punktów. Wszystkie wiersze niższe od tak dobranej granicy są usuwane bądź „doklejane" do wierszy bezpośrednio pod nimi (ewentualne „doklejenie" kropki do litery „i" lub „j"). Operacja sklejania jest realizowana wówczas, gdy są spełnione dwa warunki:

- wiersz poniżej doklejanego nie jest wyższy, niż mediana,
- odległość między klejonymi wierszami nie przekracza wysokości wiersza, dla którego jest wykonywana korekcja.

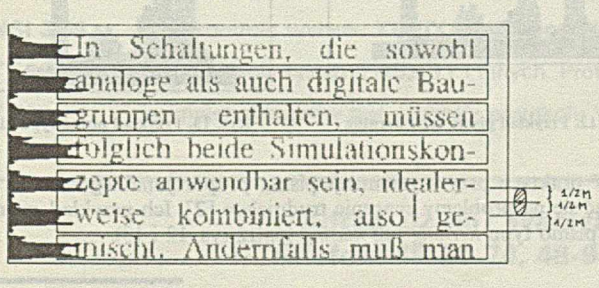


Rys. 4. Efekt wyszukiwania wierszy metodą opartą na analizie funkcji $h(\ )$

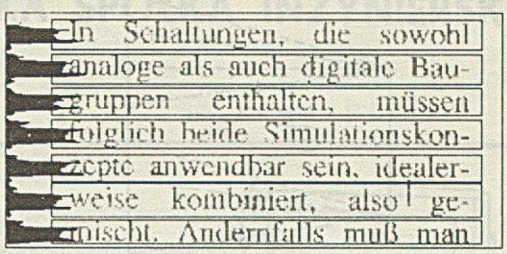
Warunki takie zostały dobrane intuicyjnie i w praktyce dają dobre rezultaty. Rysunek 4 ilustruje wynik działania algorytmu wyszukiwania wierszy.

Stosując tę samą metodę wybiera się wiersze, których wysokość jest znacznie (praktycznie ponad dwukrotnie) większa od mediany. Powodem tego zjawiska są zakłócenia, w wyniku których linie druku zlewają się. Algorytm podziału takich wierszy ilustruje rysunek 5a. Został na nim pokazany stan przed korekcją, gdzie piąty i szósty wiersz na skutek zakłócenia (pionowa kreska) nie są rozdzielone. Na osi pionowej jest wybierany punkt oddalony od podstawy sklejonej linii o odległość równą medianie wysokości wierszy. W otoczeniu tego punktu o promieniu $M/2$ szukane jest minimum funkcji $h(\ )$. Położenie tego minimum przyjmuje się za granicę podziału wiersza. Efekt powyższej operacji ilustruje rysunek 5b.

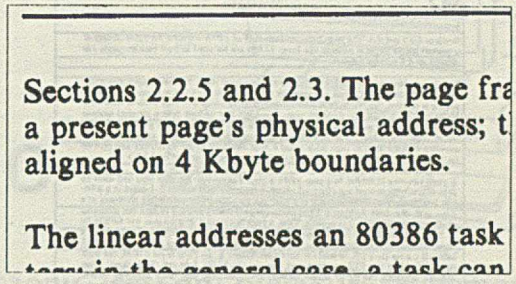
Omówiony algorytm wyszukiwania wierszy oparty na rzucie prostopadłym $h(\ )$ jest mało skuteczny, jeśli obraz binarny jest choć nieznacznie przekręcony, powodując, że wiersze nie są równoległe do postawy obrazu binarnego (patrz rysunek 6).



Rys. 5. a) Podział „zlanych" wierszy tekstu – stan przed korekcją



Rys. 5 b) Podział „zlanych" wierszy tekstu – stan po korekcji



Rys. 6. Obraz binarny dokumentu nieznacznie skręconego podczas skanowania

Dla obrotu o jeden stopień, przy interlinii 1,5 segmentacja na wiersze przestaje funkcjonować, gdyż zmienia się istotnie obraz $h(\ )$. Ilustruje to rysunek 7, w którym ramka reprezentuje układ odniesienia dla procesu segmentacji.

Piłokształtny przebieg $h(\ )$ nie ma wyraźnych przerw między wierszami. Nie jest możliwe pozbycie się tego zjawiska przez bardzo precyzyjne ulokowanie analizowanego dokumentu w skanerze, ponieważ wymagana dokładność jest nieosiągalna przy ręcznym korygowaniu położenia. Stwarza to konieczność automatycznego wypoziomowania tekstu z wymaganą dokładnością, przez odpowiednią obróbkę informacji ze skanera.

Rys. 7. Rzut prostopadły $h(\ )$ dla skręconego dokumentu

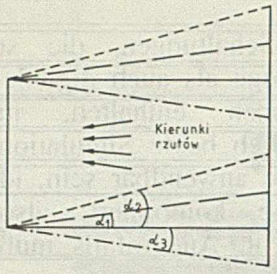
W przeprowadzonych doświadczeniach stwierdzono, iż wartości rzutu $h(\ )$ są najbardziej zróżnicowane dla tekstu ustawionego poziomo. Oznaczając przez $h_\alpha(i)$ rzut po obrocie tekstu o kąt α, za miarę tej nieregularności przyjęto współczynnik G, zdefiniowany jako:

$$G_\alpha = \sum_{i=1}^{m-1} |h_\alpha(i) - h_\alpha(i+1)| \qquad (3)$$

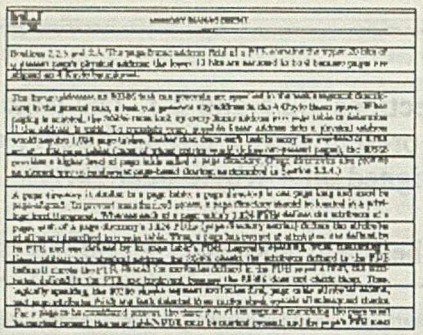
Opracowano następujący algorytm automatycznego poziomowania:

- jest wybierany obszar, na którym liczony będzie rzut $h(\ )$ (ze względu na czas przetwarzania nie jest brany pod uwagę cały dokument, lecz jego fragment $[a, b]$, $a < m$ i $b < m$),
- na tak wybranym fragmencie są liczone współczynniki G, dla różnych kątów α w otoczeniu zera stopni (rys. 8),
- następnie wyszukiwany jest największy z otrzymanych współczynników G_α, natomiast kąt, dla którego został on wyliczony, stanowi

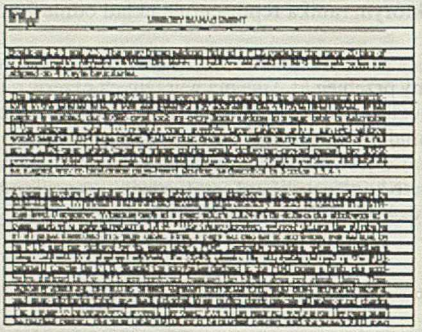
podstawę do korekcji położenia dokumentu. Wpływ automatycznego poziomowania na sprawność segmentacji ilustrują rysunki 9a i 9b.



Rys. 8. Ilustracja metody wyznaczania maksimum współczynnika G



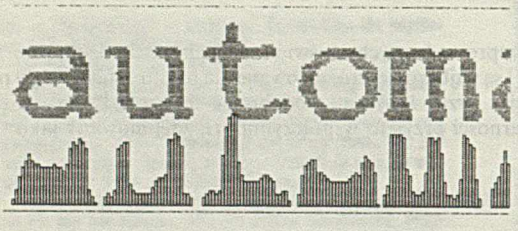
Rys. 9 a) Efekt działania algorytmu podziału na wiersze bez wcześniejszego wypoziomowania obrazu binarnego



Rys. 9 b) Efekt działania algorytmu podziału na wiersze po automatycznym wypoziomowaniu obrazu binarnego

Wydzielenie znaków

Aby określić pozycję poszczególnych znaków tekstu, konieczny jest podział wierszy na pojedyncze znaki. Realizacja tak rozumianej segmentacji pozwala przyjąć w procesie rozpoznawania, że otrzymany prostokątny fragment dokumentu zawiera dokładnie jeden znak.



Rys. 10. Przykładowa postać rzutu prostopadłego $v(\ )$ na oś poziomą

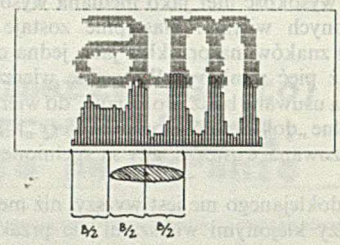
Badano skuteczność algorytmu analogicznego do użytego przy wyszukiwaniu wierszy. Rzut prostopadły na oś poziomą $v(\ )$ (rys. 10), zdefiniowano jako:

$$v(j) = \sum_{i=c}^{d} p(i,j) \qquad (4)$$

dla $j \in \{1, 2, ..., n\}$. Wartości c oraz d wskazują na pozycję analizowanego wiersza w obrazie binarnym (odpowiednio indeks pierwszej i ostatniej poziomej linii obrazu binarnego, zaliczanej do przetwarzanego wiersza). Obraz funkcji $v(\ )$ umożliwia podział wiersza na prostokąty.

Następnie wysokość prostokątów jest redukowana tak, aby możliwie dokładnie opisywały one litery. Do tego celu są wykorzystywane znów obrazy rzutów $h(\ )$ i $v(\ )$. Uzyskany obszar znaku wraz z informacją o położeniu względem podstawy przetwarzanego wiersza jest daną wejściową dla algorytmu rozpoznawania. Informacja o położeniu ma znaczenie wtedy, gdy zachodzi konieczność odróżnienia na przykład znaku „'" od „,".

Ze sklejeniami mamy do czynienia wówczas, gdy wydzielony spójny zbiór punktów w rzeczywistości reprezentuje więcej niż jeden znak. Dla druku, w którym każda litera ma tę samą szerokość, rozdzielanie połączonych liter można zrealizować podobnie, jak w przypadku połączonych wierszy. Ideę takiej korekcji pokazuje rysunek 11.



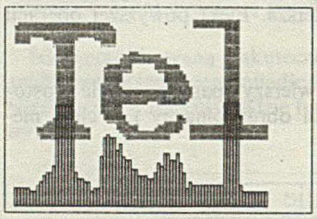
Rys. 11. Ilustracja metody rozdzielania sklejonych znaków

W tym celu przed rozpoczęciem segmentacji do programu rozpoznającego wprowadza się standardową szerokość znaku, zwaną dalej szerokością bazową.

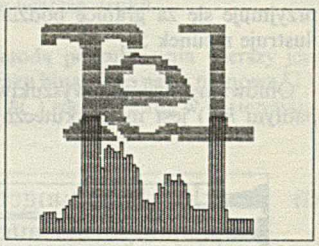
* * *

Zastosowanie powyższych metod segmentacji do tekstu o właściwościach omówionych w charakterystyce problemu dało bardzo dobre rezultaty. W przypadku pisma maszynowego o gęstości 10 cpi (10 znaków na cal) wyszukiwanie wierszy tekstu dla strony formatu A5 trwało znacznie poniżej 10 sekund, przy czym osiągnięto skuteczność 100%. Błąd podziału wierszy na znaki zależał od jakości druku i normalnie nie przekraczał 0,2%.

Dokonano również próby analizowania tekstów z pismem proporcjonalnym, gdzie litery mają różne szerokości. W druku takim występują ligatury, to znaczy sklejenia na przykład liter *fl, rt, rn*, których szerokość jest porównywalna z szerokością niektórych pojedynczych znaków (np. litery *m* czy też *W*). Metoda rozdzielania oparta na szerokości bazowej była w tym przypadku mało skuteczna. Wprowadzenie do programu wszystkich możliwych ligatur i traktowanie ich jak odrębnych znaków pozwoliło rozwiązać ten problem. Wydaje się celowym zaimplementowanie automatycznego mechanizmu do tworzenia wzorców ligatur korzystającego ze standardowego zbioru liter.



Rys. 12. Przykład pisma typu kerning

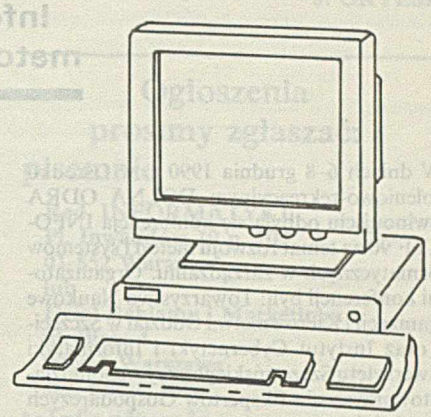


Rys. 13. Przykład pisma typu kissing

W praktyce przy segmentacji pisma proporcjonalnego można natknąć się na problemy znacznie trudniejsze [2]. Ich przykładem może być pismo typu *kerning* lub *kissing* (patrz rys. 12 i 13).

dokończenie na s. 28

POLSKA SPÓŁKA INŻYNIERSKA
Sp. z o.o.



TERMINAL AVT 180-PC

Szybki terminal znakowy AVT 180-PC przeznaczony jest do pracy w systemach wielostanowiskowych opartych na mikrokomputerach klasy IBM PC XT/AT/386, MICROVAX, PDP-11 i innych. Spotykane na rynku polskim konstrukcje zachodnie przewyższa następującymi parametrami:

- Realizuje specjalne protokoły dla systemów QNX, PC-MOS, MULTI-LINK. Dzięki temu praca na terminalu nie różni się od pracy na konsoli głównej.
- Parametry obrazu są identyczne jak w IBM PC z kartą Hercules.
- Terminal jest wyposażony w 18 generatorów znaków (9 polskich, Hercules, 2 ANSI, 5 rosyjskich i czeski). Na ekranie można jednocześnie wyświetlać wszystkie znaki z różnych generatorów co – na przykład – pozwala na napisanie części tekstu po rosyjsku, a reszty po polsku.
- Terminal pozwala na szeroki wybór układu znaków narodowych na klawiaturze. Dostępnych jest 10 układów polskich, 6 rosyjskich, amerykański, angielski, francuski, niemiecki i hiszpański.

Terminal może pracować w następujących protokołach:

PCSHADOW – pod kontrolą sytemu **MULTILINK**.
PCTERM – pod kontrolą systemu **PC-MOS**.
QNXS – pod kontrolą systemu **QNX**.
VT52+ – pod kontrolą systemu **XENIX/UNIX** i innych. Protokół rozwinięty o atrybuty znaku i sekwencje klawiatury.
ANSI 3.64 – pod kontrolą systemu **XENIX/UNIX** i innych. Protokół zgodny z normą ANSI 3,64 i ISO 6429.
VT 100 – pod kontrolą systemu **RSX 11** i innych. Protokół ograniczony do 80 znaków w linii.
VT 220 – zaimplementowany jest podzbiór protokołu VT 220.

Adres: 02-567 Warszawa, ul. Sandomierska 13
Telefon: 49-56-74, 48-97-73, Teleks: 817795 dglb pl

ZO/317/90

X Jubileuszowy INFOGRYF '90

Informatyka w zarządzaniu przedsiębiorstwem – rozwój metod i systemów informatyki stosowanej w zarządzaniu

W dniach 6–8 grudnia 1990 r. w Ośrodku szkoleniowo-rekreacyjnym DOLNA ODRA w Świnoujściu odbyła się konferencja INFOGRYF'90 na temat rozwoju metod i systemów informatycznych w zarządzaniu. Organizatorami konferencji byli: Towarzystwo Naukowe Organizacji i Kierownictwa Oddział w Szczecinie oraz Instytut Cybernetyki i Informatyki Uniwersytetu Szczecińskiego przy współudziale Stowarzyszenia Ekspertów Gospodarczych w Szczecinie.

Ponieważ był to dziesiąty, a więc jubileuszowy INFOGRYF należy przypomnieć, że konferencje o tej nazwie były organizowane w Kołobrzegu od 1970 r. w cyklu dwuletnim, ugruntowując swą pozycję w kalendarzu krajowych imprez informatycznych. Mimo kryzysowej sytuacji gospodarczej kraju nie zawiedli stali uczestnicy konferencji, gromadząc 40 uczestników z wyższych uczelni, PAN i przedsiębiorstw. INFOGRYF'90, podobnie jak wiele poprzednich konferencji miała również zasięg międzynarodowy. Oprócz pracowników nauki i praktyki gospodarczej z całego kraju, wzięli w niej udział przedstawiciele środowiska naukowego z Danii i ZSRR (Rygi), którzy z Uniwersytetem Szczecińskim współpracują na co dzień.

Konferencje INFOGRYF-u stanowią od lat główne w skali kraju formy wymiany informacji w dziedzinie zastosowań informatyki w zarządzaniu przedsiębiorstwem, pracach badawczych i metodologicznych, przyczyniając się do kształtowania na ten temat poglądów środowiska. Sprzyjają w ten sposób integracji środowiska informatyków i stymulują tempo oraz prawidłowe kierunki rozwoju informatyki w zarządzaniu przedsiębiorstwem.

Celem merytorycznym konferencji INFOGRYF'90 była prezentacja rozwiązań zastosowania informatyki w zarządzaniu przedsiębiorstwem oraz projekcja metod i systemów na przyszłość. Zgodnie z tym przedstawiono oraz przedyskutowano aktualne problemy związane z zastosowaniem informatyki w warunkach gospodarki rynkowej oraz dokonano wzajemnej wymiany doświadczeń. Tematykę konferencji ukierunkowano na stosowanie nowoczesnych metod, środków i narzędzi do budowy systemów informatycznych, systemy doradcze, opłacalność zastosowań informatyki, standaryzację procesów pomocniczych w informatycznych systemach zarządzania, problematykę sprzętu komputerowego oraz kierunki przyszłego rozwoju.

Uczestnicy konferencji otrzymali materiały konferencyjne, zawierające referaty:

- „Wpływ nowoczesnych technologii informacyjnych na rozwój systemów informatycznych w zarządzaniu" – prof. dr hab. inż. N. Bazewicz (Politechnika Wrocławska);
- „Od systemu informatycznego do systemu informacyjnego wspartego technologią komputerową" – doc. dr. hab. W. Flakiewicz (SGPiS Warszawa);
- „Badanie potrzeb informacyjnych w systemach zarządzania – zarys metodyki" – dr E. Kolbusz (Uniwersytet Szczeciński);
- „Standaryzacja procesów pomocniczych w informatycznych systemach zarządzania" – dr Barbara Łukasik-Makowska, dr M. Skwarnik (Akademia Ekonomiczna we Wrocławiu);
- „Rozwój metodyki projektowania informatycznego systemów zarządzania" – prof. dr hab. E. Niedzielska, dr M. Skwarnik (Akademia Ekonomiczna we Wrocławiu);
- „Nowe generacje systemów informatycznych w zarządzaniu" – prof. dr hab. A. Nowakowski (Uniwersytet Szczeciński);
- „Rozszerzone podejście do projektowania systemów z wykorzystaniem CASE-TOOLS" – doc. dr hab. W. Olejniczak, mgr I. Szydłowski (Uniwersytet Szczeciński);
- „Model efektywności systemów informatycznej obsługi zarządzania" – prof. dr hab. inż. P. Sienkiewicz (Akademia Obrony Narodowej);
- „Metody sztucznej inteligencji i systemy ekspertowe w zarządzaniu" – doc. dr hab. B. Stefanowicz (SGPiS Warszawa);
- „Pakiety CASE, wyzwania, możliwości, bariery" – S. Wrycza (Uniwersytet Gdański);
- „U progu nowego przełomu w technice komputerowej – zestawy wielokomputerowe z łącznością szynową jako przyszłościowa postać wielodostępowych zestawów komputerowych" – mgr inż. T. Rawiński (Politechnika Gdańska);
- „Model symulacyjny badania zdolności przepustowych bazy promowej" – mgr E. Wiśniowski (ZK – ELKON, Kołobrzeg);
- „Wybrane podejście komputerowo wspomaganego projektowania systemów informatycznych" – mgr K. Gajewski, mgr J. Żbikowski, (Uniwersytet Szczeciński);
- „Komputerowe wspomaganie podejmowania decyzji inwestycyjnych" – mgr E. Komorowska, dr B. Mażbic-Kulma, mgr A. Pogorzelec, mgr J. Rydel (IBS PAN Warszawa);
- „Nowy trend w CAMS – komputerowy lekarz systemu zarządzania" – dr inż. E. Michalewski (IBS PAN Warszawa).

Obradom przewodniczyli sekretarze naukowi konferencji: doc. dr hab. Wojciech Olejniczak i dr Edward Kolbusz.

W pierwszym dniu obrad referaty wygłaszali prelegenci spoza Szczecina. Przedmiotem dyskusji był problem rozwoju zastosowań informatyki w warunkach gospodarki rynkowej, gdy występują nowe warunki gry gospodarczej, konieczność szybkiego reagowania oraz bieżącego analizowania sytuacji zmieniającej obszar zastosowań komputerów. Typowy dotąd w kraju schemat oprogramowania (kadry, płace, gospodarka magazynowa) będzie trzeba zastąpić modelem bardziej dynamicznym. Ilość musi zmienić się w jakość. Dyskusja wykazała, że istnieje u nas duża luka informacyjna w dziedzinie oprogramowania a zakup gotowych pakietów typu CASE nie zawsze spełnia oczekiwania w polskich warunkach. Wprowadzone już zmiany w systemie zarządzania gospodarką narodową spowodowały konieczność poszukiwań nowych rozwiązań działalności produkcyjnej, sposobów obniżenia kosztów produkcji oraz jej usprawnienia, czego oczywiście nie będzie można odpowiednio szybko osiągnąć bez zastosowań informatyki.

Drugi dzień obrad był głównie dniem szczecińskim, ale nie tylko. Doc. dr hab. Wojciech Olejniczak z Uniwersytetu Szczecińskiego wraz z zespołem zaprezentował pokaz nowej, oryginalnej polskiej technologii tworzenia narzędzi informatycznych do budowy systemów informatycznych, wychodząc z fazy projektowania organizacji przedsiębiorstwa. Następnie dr inż. E. Michalewski z Instytutu Badań Systemowych PAN w Warszawie zaprezentował system DIANA-9, usprawniający istniejący system organizacyjny w przedsiębiorstwie metodą testowania obiegu informacji i przydatności ludzi na zajmowanych stanowiskach. Oba prezentowane systemy wzbudziły duże zainteresowanie uczestników konferencji. Stanowią one nowoczesne narzędzia informatyczne, przeznaczone do usprawniania lub zaprojektowania od nowa organizacji przedsiębiorstwa. Problematyka ta jest szczególnie aktualna w obecnych warunkach gospodarczych kraju i może być pomocna zwłaszcza w praktycznym projektowaniu organizacji.

Przedmiotem dyskusji był także problem standaryzacji systemów powielarnych. Użytkownik chce otrzymać produkt (system) przyjazny i prosty w obsłudze, co w znacznym stopniu zapewnić może standaryzacja.

W referatach i dyskusjach wyrażono opinię na temat dalszego rozwoju systemów doradczych oraz małą efektywność obecnych zastosowań informatyki w zarządzaniu.

Obradom towarzyszyły imprezy okolicznoś-

ciowe, takie jak „Mikołaj" oraz „Okrągły stół", przy którym w trzecim dniu obrad toczyła się dyskusja głównie na temat zastosowań informatyki. W dyskusji tej podkreślano zwłaszcza, ograniczenia sprzętowe. Chodziło przede wszystkim o średnie i duże komputery oraz oprogramowanie dla dużych przedsiębiorstw, jak też zakres możliwości zastosowań w technice mikrokomputerowej. Konferencja odbywała się w atmosferze zwiększonego zainteresowania efektywnością zastosowań i potwierdziła swoją przydatność dla praktyki gospodarczej.

Wydaje się, że wyniki konferencji INFOGRYF'90 w znacznym stopniu mogą przyczynić się do ukierunkowania dalszych prac nad zastosowaniem informatyki w zarządzaniu przedsiębiorstwem. Zagadnienia te stanowiły nie tylko przedmiot ożywionej dyskusji w czasie konferencji, lecz także są rozwiązywane w codziennym warsztacie pracy informatyka i użytkownika systemu w konkretnym przedsiębiorstwie.

Na podstawie dyskusji sformułowano następujące wnioski:

1. Podniesienie efektywności ekonomicznej zastosowań informatyki w zarządzaniu przedsiębiorstwem, obniżenie kosztów produkcji, zmniejszenie strat wymaga rozwoju kultury informatycznej pracowników odpowiedzialnych za realizację zadań w przedsiębiorstwie.

2. Istnieje konieczność opracowania długofalowego programu rozwoju zastosowań informatyki w warunkach gospodarki rynkowej.

3. Orientowanie się przedsiębiorstw na rynek krajowy i zagraniczny wymaga gromadzenia koniecznych informacji o aktualnych oraz prognozowanych potrzebach, co nie jest możliwe bez zorganizowania banku informacji o metodach i modelach w warunkach komputerowych.

4. Działanie przedsiębiorstw na rynku zmusza do śledzenia trendów i tendencji oraz przedsiębiorczości w nawiązywaniu kontaktów wychodzenia z nowymi produktami na rynki wewnętrzny i zagraniczny. Systemy informatyczne i oprogramowanie muszą stać się towarem jak każdy inny towar. Nie można tolerować piractwa oprogramowania.

5. Prezentowane na konferencji INFOGRYF'90 systemy informatyczne powinny zostać włączone do problemu resortowego RI-14 celem ich wykorzystania w dydaktyce szkoły wyższej na zajęciach z informatyki bądź zarządzania przedsiębiorstwami.

Należy podkreślić, że przyjazna i kameralna atmosfera dyskusji jaką zapewnili prowadzący obrady pozwoliła uczestnikom na wyjątkowo swobodne wypowiadanie myśli i problemów w obszarze tematyki konferencji.

J. ORYLSKA

Ogłoszenia prosimy zgłaszać:

pisemnie

Red. INFORMATYKI
Pl. Inwalidów 10 p. 104
01-552 Warszawa
lub
Dział Reklamy i Marketingu
ul. Biała 4
00-950 Warszawa
skr. poczt. 1004

telefonicznie

nr 39-14-34
lub 20-31-16

telefaksem

nr 20-31-16

teleksem

814550 SIGMA PL
z zaznaczeniem
– Red. INFORMATYKI

MODEMY

□ homologowane, 2400/1200/300 b/s, na łącza komutowane i dzierżawione, wolnostojące i na karcie do PC, także zgodne ze standardem Hayes

oraz

□ konwertery 9600 b/s baseband
□ konwertery RS232 – pętla prądowa
□ tester V24/RS232 – pocket size
□ programator EPROM 2716-27512 oraz mikroprocesorów jednoukładowych rodziny MCS48, MCS51
□ interfejsy pomiarowe IEC 625 (IEEE 488) do PC
□ pakiet transmisji synchronicznej BSC do PC
□ oprogramowanie emulacyjne stacji IBM 3780
□ koncentrator stanowisk operatorskich do MERY 9150
□ adapter telekomunikacyjny do ODRY 1305

oferuje

SWEDEX UNIVERSAL Co. LTD
ul. Wspólna 1
05-075 Wesoła k. Warszawy
tel. (0-22) 153-365
teleks 813935 swedx pl

ZO/236/90-A

PROMOCYJNA 80% OBNIŻKA CEN!

LOGOTEC® Engineering

Oferujemy

licencjonowane, polskie wersje językowe zachodnioniemieckich systemów wspomagania projektowania

LOGOCAD – mechanika **ELOCAD** – elektrotechnika

oprogramowanie pomocnicze

pakiety ponad 100 polskich (PN) i niemieckich (DIN) norm mechanicznych (śruby, nakrętki, podkładki, pierścienie, zawleczki, sworznie, sprężyny, łożyska toczne itp.)

inżynierskie programy obliczeniowe:

(wały, belki, ramy, kraty, ruszty, płyty, powłoki, płaski stan naprężeń i płaski stan odkształceń, przekroje cienko- i pełnościenne, obliczenia i optymalizacja kół zębatych)

programy do obsługi baz danych

UNIP2 – uniwersalny generator wydawnictw i programów standardu dBase i Clipper
nagrodzony MikroLaurem '88 i '90
ARCHIW – archiwum rysunków i projektów
MOL(książkowy) – do kompleksowej obsługi biblioteki i wypożyczalni książkowej

doradztwo i szkolenia

z zakresu zastosowania naszych systemów CAD, kursy obsługi naszego oprogramowania, pomoc we wdrożeniu i eksploatacji programów

sprzęt

kompletne stanowiska dla naszych systemów, digitizery, monitory i karty graficzne dla różnych systemów CAD.

Jesteśmy do państwa dyspozycji

LOGOTEC ENGINEERING
Zakład Rozwoju Systemów CAD
Nasz nowy adres: 40-028 KATOWICE, ul. Francuska 70
tel.: 57-46-26, 57-23-76, 51-17-47, teleks 0312606

0/21/90

Wyniki konkursu prac magisterskich

Zakończył się VII Ogólnopolski Konkurs PTI na najlepsze prace magisterskie z informatyki, organizowany przez Oddział Dolnośląski naszego Towarzystwa.

Na posiedzeniu 11 grudnia 1990 roku w Centrum Obliczeniowym Politechniki Wrocławskiej Komisja Konkursowa w składzie: doc. dr inż. Czesław Daniłowicz (przewodniczący), doc. dr hab. Piotr Dembiński, doc. dr hab. Zbigniew Huzar, doc. dr hab. Maciej Sysło, prof. dr inż. Jan Zabrodzki i dr inż. Zbigniew Szpunar (sekretarz), uwzględniając opinie recenzentów 25 prac konkursowych, po dyskusji ustaliła następujące rozstrzygnięcie konkursu:

P i e r w s z e j nagrody nie przyznano.

D r u g ą nagrodę w wysokości zł 750000.- otrzymuje

Andrzej Piotr Szyndrowski za pracę pt. *Algorytmy automatycznego generowania krzyżówek. System generowania krzyżówek APIS,* wykonaną w Uniwersytecie Wrocławskim (Wydział Matematyki, Fizyki i Chemii, Instytut Informatyki; promotor dr Ewa Gurbiel).

Dwie równorzędne t r z e c i e nagrody po zł 600000.- otrzymują:

Andrzej Leonarski i **Sławomir Wernikowski** za pracę pt. *Projekt i realizacja translatora języka TTCN na maszynę IBM PC,* wykonaną w Politechnice Wrocławskiej (Wydział Informatyki i Zarządzania, Centrum Obliczeniowe; promotor dr inż. Iwona Dubielewicz) oraz

Radosław Pruchnik i **Mariusz Rudnicki** za pracę pt. *System SORTING. Komputerowe wspomaganie nauczania o sortowaniu,* wykonaną w Uniwersytecie Wrocławskim (Wydział Matematyki, Fizyki i Chemii, Instytut Informatyki; promotor doc. dr hab. Maciej M. Sysło).

Trzy równorzędne w y r ó ż n i e n i a po zł 350000.- otrzymują:

Tomasz Kalinowski za pracę pt. *Projekt, realizacja i zastosowanie pakietu wspomagającego budowę programów uruchomieniowych,* wykonaną w Politechnice Warszawskiej (Wydział Elektroniki, Instytut Informatyki; promotor mgr inż. Tomasz Byzia),

Zdzisław Król za pracę pt. *Generowanie wysokiej jakości obrazów komputerowych za pomocą techniki śledzenia promieni,* wykonaną w Uniwersytecie Warszawskim (Wydział Matematyki, Informatyki i Mechaniki, Instytut Informatyki; promotor dr Michał Jankowski) oraz

Małgorzata Włodarczak i **Sławomir Włodarczak** za pracę pt. *System LISTDATA. Komputerowe wspomaganie nauczania o strukturach listowych,* wykonaną w Uniwersytecie Wrocławskim (Wydział Matematyki, Fizyki i Chemii, Instytut Informatyki; promotor doc. dr hab. Maciej M. Sysło).

Sekretarz Komisji Konkursowej
Zbigniew Szpunar

Czwarta Szkoła Wiosenna PTI

W dniach 20-24 maja br. odbędzie się w Świnoujściu Czwarta Szkoła Wiosenna PTI. To spotkanie informatyków będzie poświęcone sieciom komputerowym. W odróżnieniu od innych spotkań na ten temat, podstawą rozważań staną się problemy zastosowań sieci komputerowych w praktyce i wynikające z tego wnioski do użytkowania, projektowania, programowania i wdrażania systemów informacyjnych. Dokona się próby przeglądu większości rodzajów sieci najpopularniejszych i mających najwięcej zastosowań w Polsce.

Będą przedstawione doświadczenia z użytkowania systemu operacyjnego Xenix. Podzielą się nimi koledzy z Górnośląskiego Oddziału PTI. Doświadczenia z użytkowania systemu Novell staną się tematem wystąpień kolegów z Politechniki Poznańskiej.

Pojawiająca się na naszym rynku komputerowym firma Oracle przedstawi swe produkty w referatach wygłaszanych przez przedstawicieli firmy. Tematyka tych wystąpień będzie skoncentrowana na rozwiązaniach sieciowych firmy.

Rozległe sieci komputerowe — to tematyka dość abstrakcyjna w warunkach polskich, ale ostatnie zmiany otwierają nowe perspektywy przed tym rodzajem zastosowań informatyki. Doświadczenia, rozwiązania i perspektywy sieci KASK opartej na sprzęcie i oprogramowaniu IBM będą tematem referatów kolegów z Centrum Obliczeniowego Politechniki Wrocławskiej.

Przedstawiciel IBM opisze rozwiązania sieciowe oferowane przez tę firmę. Pragnie ona ponownie znaleźć się na polskim rynku, co powinno wynikać z pozycji, którą zajmuje na rynku światowym.

Tradycyjnie już na program szkoły złożą się sesje przed- i popołudniowe; przed południem — trzygodzinne referaty programowe uzupełniane godzinną dyskusją, po południu — po dwa referaty 45-minutowe z dyskusją, stanowiące uzupełnienie porannej tematyki. Mamy nadzieję na pokazy sprzętowe rozwiązań omawianych w referatach oraz na owocne dyskusje wieczorne.

Organizatorzy szkoły uprzejmie dziękują firmom IBM i Oracle, które są sponsorami szkoły.

Wszelkie informacje można uzyskać w Kole PTI w Szczecinie, tel. 760-31, w. 317 i 319.

Zdzisław Szyjewski

Instytut Podstaw Informatyki PAN w Warszawie organizuje w dniach 18–23 maja 1992 r. w Kazimierzu Dolnym II Krajową Konferencję Naukową pn.

GRAFIKA I WIDZENIE MASZYNOWE '92

Tematyka Konferencji: języki i bazowe systemy graficzne, bazy danych graficznych, systemy przetwarzania obrazów, systemy maszynowego widzenia, algorytmy dla grafiki, przetwarzanie obrazów i maszynowego widzenia, nowe rozwiązania sprzętowe, psychofizjologia widzenia, fotogrametria, grafika komputerowa w sztuce, wybrane zastosowania itp.

W programie przewidziano kilka wykładów przez zaproszonych specjalistów zagranicznych i krajowych oraz prezentacje zgłoszonych prac. Termin nadsyłania zgłoszeń uczestnictwa upływa 30 lipca, a referatów – 30 października bieżącego roku.

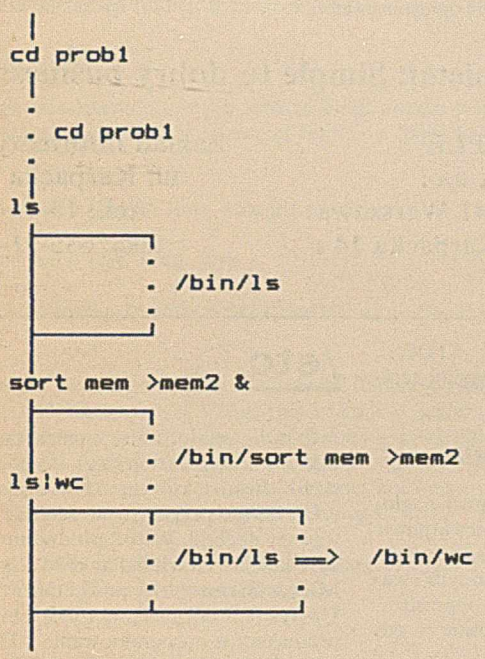
Bliższe dane na temat Konferencji można uzyskać u jej organizatorów pod adresem: **Konferencja GiW'92, IPI PAN, ul. Polna 18/20, 00-625 Warszawa, tel. 25-18-97.**

Procesy, procesy...

(w shellu Bourne'a)

Jeżeli chcemy używać shella w sposób bardziej wyrafinowany niż to pokazują najprostsze podręcznikowe przykłady, warto wiedzieć, jak polecenia shellowe odwzorowują się na procesy UNIX-owe. Jedno polecenie shella może powołać jeden lub więcej procesów, a może też nie powołać żadnego. Dla przykładu rozważmy następujący ciąg poleceń:

```
cd prob1
ls
sort men >mem2 &
ls|wc
```



Rysunek przedstawia realizację tych poleceń z zaznaczeniem nitek poszczególnych procesów. Główna nitka jest nitką procesu shella: jest inicjowana w momencie zgłoszenia się użytkownika i biegnie aż do naciśnięcia przez tegoż użytkownika znaku ^D. Ta nitka czyta i rozpoznaje polecenia, a następnie inicjuje wykonanie poleceń lub wprost je wykonuje.

Pierwszym poleceniem jest **cd:** należy ono do tzw. poleceń specjalnych. Polecenia specjalne wykonuje sam shell, nie powołuje się dla nich żadnego procesu.

Polecenie **ls** jest wykonywane przez program **/bin/ls,** wywoływany jako oddzielny proces (za pomocą pary ekstrakodów **fork/exec**). Gdy wykonuje się **/bin/ls**, proces shella czeka.

Podobnie, polecenie **sort** jest realizowane przez program **/bin/sort** jako oddzielny proces, ale proces shella nie czeka na jego zakończenie, gdyż jest to polecenie z żądaniem wykonania w tle (wskazuje na to ogranicznik &).

Ostatnie polecenie jest wykonywane przez dwa programy **/bin/ls** i **/bin/wc** biegnące jako dwa współbieżne procesy, połączone potokiem. Segmenty łącza potokowego są zawsze realizowane przez oddzielne procesy, nawet jeśli któryś z nich zawiera wyłącznie polecenia specjalne.

Możemy wymusić powołanie nowego wcielenia shella do wykonania wskazanej sekwencji poleceń używając nawiasów okrągłych ( ). Należy pamiętać, że wykonanie w oddzielnym procesie shella niesie określone konsekwencje semantyczne. Weźmy np. pod uwagę polecenie

```
{cd prob2; HOME=prob2; medit;}
```

oraz

```
(cd prob2; HOME=prob2; medit)
```

Po wykonaniu pierwszego z tych poleceń skorowidzem roboczym będzie **prob2** i zmienna **HOME** otrzyma wartość **prob2**, natomiast po wykonaniu drugiego – ani skorowidz roboczy, ani zmienna **HOME** nie ulegną zmianie. Dlaczego? Bo dla sekwencji poleceń w nawiasach okrągłych powołuje się oddzielny proces shella. W tym oddzielnym procesie co prawda zmieni się analogicznie skorowidz roboczy i zmienna **HOME**, ale po wykonaniu całej sekwencji ów proces zniknie.

Weźmy z kolei pod uwagę polecenie strukturalne np. **for:**

```
for i in $*
  do
    grep $i*
done
```

Sterowanie pętlą **for** jest realizowane przez bieżący proces shella, a tylko ewentualnie dla poleceń wewnętrznych pętli powołuje się oddzielne procesy zgodnie z ich wymaganiami. W naszym przypadku w kolejnych cyklach pętli będzie pojawiał się każdorazowo tylko jeden oddzielny proces dla realizacji polecenia **grep**.

Podobnie realizowane są przez bieżący proces shella polecenia **while, until, case** oraz oczywiście zgrupowanie za pomocą nawiasów {}. Następujące polecenie:

```
for i in $*
  do
    case $i in
      *.c) sources=$sources' '$i ;;
      *.o) objects=$objects' '$i ;;
      *)   others=$others' '$i ;;
    esac
done
```

nie wymaga zatem żadnego dodatkowego procesu; wykona się całkowicie w procesie bieżącego shella.

Jeżeli dla polecenia strukturalnego wyspecyfikujemy skierowanie wejścia lub wyjścia, np.:

```
for ...
  do
    ...
done>outfile
```

do wykonania takiego polecenia jednak zostanie powołany nowy proces shella. Dzieje się tak dlatego, że nie można przedefiniowywać wejść-wyjść bieżącego shella; zatem aby uzyskać pożądany efekt skierowania, powołuje się w takich przypadkach oddzielny proces i oczywiście pociąga to za sobą odpowiednie konsekwencje semantyczne.

Wykonanie omawianych poleceń strukturalnych w tle również nie może być zrealizowane w bieżącym procesie shella i będzie wymagało

powołania nowego procesu. Nie znaczy to, że np. wykonanie polecenia **for** w tle ze skierowaniem we-wy, jak niżej,

```
for ...
  do
    ...
done >outfile &
```

będzie wymagało aż dwóch dodatkowych procesów: procesy shella powołuje się tylko w takiej ilości jaka jest niezbędna do prawidłowego wykonania polecenia.

Wiedza o tym jak odwzorowują się polecenia shella na procesy może się przydać w przypadkach, kiedy chodzi nam o uzyskanie możliwie efektywnego skryptu, oraz – co starałem się pokazać – może wyjaśnić istotne problemy semantyczne tego narzędzia.

Na zakończenie zagadka: ile procesów zostanie użytych do wykonania poniższego polecenia?

```
({set;} 2>&1 | cat >outfile)&
```

W razie kłopotów z rozwiązaniem proszę zwrócić się do najbliższej instalacji UNIX-a pamiętając tylko o tym, że należy tak postawić pytanie, aby UNIX je zrozumiał.

JAN WALASEK

SIMPLE® **Sp. z o.o.**

to organizacja + informatyka
to największy w Polsce
Software House

Oferujemy wszystkie dostępne na rynkach zachodnich pakiety oprogramowania licencjonowanego po bardzo niskich cenach. I tak np.:

Microsoft
- Quick C wersja 2.0 – 1,5 mln zł
- Windows wersja 3.0 – 2,5 mln zł

Borland International
- Turbo PASCAL wersja 6.0 Professional – 3,9 mln zł
- Turbo C wersja 3.0 Professional – 3,9 mln zł

Peter Norton Computing
- Norton Commander wersja 3.0 – 2,1 mln zł
- Norton Utilities wersja 5.0 – 2,5 mln zł

Nantucket
- Clipper wersja 5.0 –7,5 mln zł

Pełny katalog oprogramowania (aktualizowany co miesiąc) oferujemy w cenie 50 tys. zł lub udostępniamy w siedzibie firmy. Realizujemy każde zamówienie w terminie do trzech tygodni.

Oferujemy poza tym własne oprogramowanie użytkowe przedsiębiorstw, przeznaczone do pracy w sieci firmy Novell, opracowane na podstawie oryginalnej bazy danych SBase, nowoczesne narzędzia – język C, pakiet dialogu Vermont Views. Zapewniamy szkolenie i wersje źródłowe oprogramowania.

Pamiętaj: Simple to dobry business!

SIMPLE
Sp. z o.o.
04-541 Warszawa
ul. Karpacka 14 c

Zakład Informatyki
ul. Karpacka 12
tel.: 15-49-83
faks: 635-91-66

SO/84/91

/pomysły

Sposób na deklaracje z prototypami w ANSI C

Korzyści płynące z używania prototypów w deklaracjach funkcji (por. sąsiadujący artykuł W. Majerskiego o ANSI C) są bezdyskusyjne dla każdego, komu choć raz zdarzyło się wywołać funkcję inaczej niż została zadeklarowana. Problem jednak w tym, że wciąż jeszcze wiele UNIX-owych komplikatorów C albo pochodzi z czasów sprzed normy ANSI, albo nie w pełni się z nią zgadza. Tak więc programista mający szczęście pracować z kompilatorem ANSI C musi liczyć się z trudnościami przenoszenia programów, w których używał deklaracji z prototypami.

Oczywiście nie jest to wielki problem: wystarczy w wierszu deklaracji ująć w komentarz wszystko, co mówi o liczbie i typach argumentów, np.:

```
extern long fun (/* int, long, int */);
```

ale jest to metoda tyleż prosta, co czasochłonna – powrót do pracy z kompilatorem ANSI C wymaga usunięcia komentarzy.

Znacznie lepszy efekt daje użycie preprocesora. Jeśli na początku programu umieścimy następującą klauzulę:

```
#ifdef __STDC__
#if __STDC__ == 1
#define __(p) p
#else
#define __(p) ()
#endif
#else
#define __(p) ()
#endif
```

(kompilator ANSI C predefiniuje __STDC__ jako 1), każdą deklarację z prototypami zapiszemy odpowiednio w postaci:

```
extern long fun __((int, long, int));
```

i zachowamy tradycyjną formę definicji nagłówka funkcji, to nasz program niewątpliwie zyska na przenośności. W przypadku większych programów wszystkie tak zmodyfikowane deklaracje z prototypami wraz z poprzedzającą je klauzulą warto przechowywać w oddzielnym pliku nagłówkowym.

Strzeżonego UNIX strzeże

Pod takim zbiorczym tytułem zamierzamy publikować artykuły o mocnych i słabych stronach UNIX-owych mechanizmów obronnych, a więc między innymi o wirusach, robalach, koniach Trojańskich i o sposobach walki z nimi. Z góry zapraszamy wszystkich do współredagowania tego cyklu i prosimy o listy. A póki co, dla mniej wtajemniczonych propozycja ćwiczenia wprowadzającego w temat: napisać możliwie najkrótszy program samoreprodukujący, tj. taki, który – skompilowany i wykonany – produkuje wynik w postaci swego własnego kodu źródłowego. Zadanie jest znane z literatury, życzę dobrej zabawy, a mniej cierpliwym polecam krótki artykuł Kena Thompsona *„Reflections on Trusting Trust"* w *Communications of the ACM* z sierpnia 1984.

/etc.

Ostatni zeszłoroczny numer miesięcznika *UNIXWORLD* zamieszcza listę amerykańskch firm, którym interesy związane z UNIX-em przyniosły w zeszłym roku największy dochód. Wśród producentów sprzętu są to kolejno **Hewlett-Packard** (2.8 mld), **Sun Microsystems** (2.75 mld), **DEC** (1.6 mld), **Unisys** (1.0 mld) i **IBM** (0.99 mld), a wśród producentów oprogramowania – **Dracle** (250 mln), **Informix** (177 mln), **SCO** (100 mln) i **Interactive** (60 mln, wszystkie kwoty w dolarach). Dane są szacunkowe i dyskusyjne, np. według innych źródeł firma Sun wyprzedza w tej klasyfikacji HP (choć jest mniejsza, jej dochody w całości wiążą się z UNIX-em, podczas gdy dochody HP tylko w 17%). Lista nie uwzględnia AT&T UNIX System Laboratories ze względu na brak danych.

* * *

Z kolei w styczniowym numerze *UNIXWORLD* znajdujemy listę najlepszych UNIX-owych produktów sprzętowych i programowych zeszłego roku. W dziedzinie sprzętu redakcja wyróżniła m.in. IBM-owską rodzinę komputerów **RISC System/6000** i odporny na uszkodzenia system **Integrity S2** firmy Tandem, a w dziedzinie oprogramowania m.in. inteligentny arkusz kalkulacyjny **20/20 Real Time Spreadsheet** firmy Access Technology oraz DOS-owo/UNIX-owy zintegrowany pakiet sieciowy **Rhapsody System** autorstwa AT&T.

JACEK SURMA

O standardzie ANSI języka C (2)

Składnia i semantyka

Najistotniejszą i chyba jedyną zmianą w składni języka C jest wprowadzenie w standardzie ANSI pojęcia **prototypu** funkcji będącego rozszerzeniem deklaracji funkcji i jednocześnie alternatywną formą definicji nagłówka funkcji. Zgodnie z podręcznikiem Kernighana i Ritchiego deklaracja funkcji niesie jedynie informację o typie wartości zwracanej przez funkcję, nie mówiąc nic o liczbie i typach jej argumentów. Wprowadzenie prototypu w deklaracji funkcji specyfikuje również liczbę i typ argumentów funkcji. Obok deklaracji funkcji z użyciem prototypu norma ANSI dopuszcza również deklarację funkcji według starej składni.

Przykład 1

Wystąpienie deklaracji funkcji w postaci

```
extern long lseek ( );
```

daje jedynie informację o tym, że funkcja **lseek** zwraca wartość typu **long.** Wystąpienie deklaracji tej samej funkcji z użyciem prototypu w formie

```
extern long lseek (int fd, long pos, int mode);
lub
extern long lseek (int, long, int);
```

informuje kompilator, że funkcja **lseek** ma trzy argumenty o typach **int, long** i **int.**

Wystąpienie nazw argumentów w deklaracji funkcji nie jest obligatoryjne, a ma jedynie znaczenie komentujące. Użycie deklaracji funkcji z użyciem prototypu pozwala kompilatorowi na sprawdzenie poprawności wywołania funkcji ze względu na liczbę i typ argumentów. Co więcej, umożliwia kompilatorowi automatyczną konwersję typu argumentu aktualnego na typ argumentu formalnego.

Standard ANSI umożliwia również deklarację z użyciem prototypu funkcji o zmiennej liczbie argumentów.

Przykład 2

Wystąpienie deklaracji funkcji w postaci

```
extern int sprintf (char *str, const char *patt, ...);
lub
extern int sprintf (char *, const char *, ...);
```

deklaruje **sprintf** jako funkcję z co najmniej dwoma argumentami typu wskaźnik do **char**, po których wystąpić może 0 lub więcej argumentów dowolnego typu.

Jak wspomnieliśmy na wstępie prototyp funkcji umożliwia alternatywną formę definicji nagłówka funkcji.

Przykład 3

Definicja funkcji postaci

```
int fun(n)
int n;
char *cp;
{/*...*/}
```

jest równoważna definicji

```
int fun(int n, char *cp)
{/*...*/}
```

lub w przypadku, gdy **fun** jest funkcją o zmiennej (2 lub więcej) liczbie argumentów, definicji

```
int fun(int n, char *cp, ...)
{/*...*/}
```

Wśród wielu, czasami niewielkich i mało istotnych zmian i uściśleń dotyczących semantyki ANSI C warto wspomnieć o trzech.

Pierwsza dotyczy typów **struct** i **union.** Różne typy struktury lub unii mogą zawierać składowe o takich samych nazwach. Norma ANSI zniosła warunek, żeby składowe o identycznych nazwach miały identyczne przesunięcie (ang. *offset*) względem początku struktury-unii.

Druga zmiana dotyczy wyrażeń typu **struct** i **union.** Wyrażenia takich typów mogą być argumentami operatora przypisania = pod warunkiem, że obydwa wyrażenia są tego samego typu, strukturą lub unią. Wyrażenie typu **struct** i **union** może być także argumentem funkcji; co więcej, funkcja może zwracać wartość takiego typu.

Norma ANSI definiuje dozwolony typ wyrażenia w instrukcji **switch.** Wyrażenie takie, po konwersji automatycznej lub wymuszonej operatorem rzutowania, musi być typu **int** lub **long** ze znakiem lub bez. W takim przypadku typ każdego stałego wyrażenia występującego po **case** jest automatycznie przekształcany na typ wyrażenia występującego w instrukcji **switch.**

Kompilator

Standard ANSI specyfikuje wymagania stawiane kompilatorom języka C. Kompilator powinien być w stanie przetłumaczyć plik źródłowy napisany w języku C, w którym występują konstrukcje językowe zawierające co najmniej:

- 15 poziomów zagnieżdżenia instrukcji złożonych, instrukcji pętli lub instrukcji wyboru,
- 6 poziomów zagnieżdżenia kompilacji warunkowej,
- 12 operatorów typotwórczych (*, [], ()) w deklaracjach lub definicjach,
- 127 poziomów zagnieżdżenia nawiasów w wyrażeniach,
- 511 identyfikatorów zmiennych nielokalnych dla pliku,
- 127 identyfikatorów definiowanych (deklarowanych) w pojedyńczym bloku,
- 1024 makrodefinicji,
- 31 argumentów w definicji lub wywołaniu funkcji,
- 31 argumentów w makrodefinicji,
- 509 znaków w linii pliku źródłowego,
- 509 znaków w stałej tekstowej (po ewentualnym sklejeniu),
- 8 poziomów zagnieżdżenia dyrektywy # **include,**
- 257 etykiet **case** w instrukcji **switch.**

Norma zakłada, że identyfikatory zmiennych lokalnych w pliku są różne, jeżeli różnią się którymkolwiek znakiem na pierwszych 31 pozycjach. Identyfikatory zmiennych nielokalnych w pliku różnią się, jeżeli różnią się którymkolwiek znakiem na pierwszych 6 pozycjach.

Wymienione wyżej warunki są minimalnymi wymaganiami; norma ANSI zaleca, żeby tam gdzie to tylko możliwe nie narzucać żadnych ograniczeń.

Standard ANSI specyfikuje kolejność czynności wykonywanych przez kompilator. Nie oznacza to bynajmniej, że każdej czynności musi

UNIX jest zastrzeżonym znakiem towarowym AT & T.

odpowiadać jeden przebieg kompilatora. Kompilacja pliku źródłowego ma przebiegać w następujących fazach:

1. Znaki (ciągi znaków) pliku źródłowego zostają odwzorowane na reprezentacje znaków alfabetu języka C (trigrafy zostają zamienione na reprezentacje pojedynczych znaków).
2. Plik źródłowy zostaje podzielony na logiczne wiersze w ten sposób, że sekwencje \ i znaku nowej linii zostają pominięte. Niepusty plik źródłowy musi kończyć się znakiem nowej linii nie poprzedzonej znakiem \.
3. Plik źródłowy zostaje przekształcony na ciąg atomów leksykalnych (w sensie preprocesora) i „białych znaków" (włączając komentarze). Każdy komentarz zostaje zastąpiony przez jeden znak odstępu.
4. Wykonywane są dyrektywy preprocesora i rozwinięcia odwołań do makrodefinicji. W przypadku wystąpienia dyrektywy # **include** wyżej wymienione fazy są rekurencyjnie wykonywane dla pliku włączanego tą dyrektywą.
5. Sekwencje oznaczające znaki specjalne (ang. escape sequence) np. \n, \t, \033 występujące w stałych znakowych i stałych tekstowych zostają zastąpione przez te znaki.
6. Stałe tekstowe występujące bezpośrednio po sobie zostają sklejone.
7. Plik źródłowy traktowany teraz jako ciąg atomów leksykalnych zostaje poddany analizie składniowej i semantycznej oraz generacji kodu.

(cdn)

WŁADYSŁAW MAJERSKI

KARTY PRZETWORNIKÓW
ANALOGOWO-CYFROWYCH
do komputerów IBM XT/AT/386
Podzespoły ANALOG DEVICES
MAXIM BURR BROWN

- 12, 14, 22 bity
- izolacja galwaniczna
- pamięć buforowa
- praca „w tle"

próbkowanie do 1 MHz!!!
KARTY PROCESORÓW SYGNAŁOWYCH
TMS320C25 MOTOROLA 56001 AT&T DSP32C
Bogate oprogramowanie
Kompleksowa obsługa zamówień
Serwis ★ Gwarancja ★ Pomoc w instalacji

PRACOWNIA USŁUG INFORMATYCZNYCH
I KOMPUTEROWYCH 'CONVERT'
50-412 Wrocław, ul. Mazowiecka 17,
telefon 300-11 w. 210, 44-85-66 telefaks 44-85-66
O/4/91

FOR rok = 1 TO 5
Studiowanie
NEXT rok

dokończenie z III strony okładki

Pozostaje nam więc liczyć na najzdolniejszych, którzy w związku z wyrównywaniem kursu złotego do dolara, nie będą mieli ochoty emigrować i w kraju staną się liderami nowych projektów. Zaś absolwentom podobnym do opisanego na początku nie wróżę kariery w informatyce. Tyle słów od nauczyciela akademickiego z szesnastoletnim stażem pracy na uczelni – na szczęście z przerwami na zdobycie nieco praktyki.

WACŁAW ISZKOWSKI

P.S. Zbitki poglądów i spostrzeżeń tutaj prezentowanych nie są oficjalnym stanowiskiem Redakcji ani żadnej innej organizacji. Ich prezentacja ma służyć pobudzeniu dyskusji na dany temat, nawet przez wzburzenie moich ewentualnych adwersarzy.

Posiadasz maszyny

typu IBM, serii SM, PDP-11, MERA400...,

pracujące

pod nadzorem systemów operacyjnych

UNIX, XENIX, RSX, PC-MOS, MULTILINK, CONCURRENT PCDOS, QNX, CROOK...

To z całą pewnością są ci potrzebne

TERMINALE ALFANUMERYCZNE
MT220

Skontaktuj się z nami.

MICRONET

Zakłady Elektroniczne
"MICRONET"
81-836 Sopot ul. Krasickiego 9
tel. 51-13-17 tlx 051-2876 mcrnt

0/17/90

ZAPRASZAMY

TERA Spółka z o. o.

Przedsiębiorstwo Popierania Postępu TERA Spółka z o.o. uprzejmie informuje, że posiadając kilkuletnie doświadczenie w instalacji systemów wspomagających zarządzanie

o f e r u j e :

- opracowanie projektów systemów,
- optymalny kosztowo i rozwojowo dobór sprzętu i oprogramowania,
- instalację systemu u klienta,
- bezpłatnie przez rok konserwację oprogramowania oraz serwis sprzętu wraz z doradztwem techniczno-eksploatacyjnym,
- szybką dostawę uzupełnień konfiguracji lub sprzętu mikrokomputerowego niezależnego od systemu, w tym mikrokomputerów ALR firmy Wearnes Technology.

Wszelkie dodatkowe informacje uzyskają Państwo codziennie oprócz niedziel w Biurze Handlowym
40–025 Katowice, ul. 27 Stycznia 31/17
tel. (faks): 51-61-65, teleks: 315448 tera pl

PAMIĘTAJ! Instalacje XENIX/NOVELL/PC MOS 386 oraz serwis to nasza specjalność – ZAPRASZAMY, ponieważ czterech lat doświadczeń nigdzie nie kupisz.

0/19/90

ul. Postępu 11
02-676 Warszawa
tel.: 43-26-36, 43-02-48
teleks: 812488,
telefaks: 43-45-21

Unitronex Corp. polsko-amerykańska spółka działająca na polskim rynku od 17 lat oferuje sprzęt komputerowy renomowanych producentów amerykańskich:

PRIME COMPUTER, INC.	– 32 bitowe komputery oparte na procesorach RISC i pracujące pod kontrolą systemu operacyjnego będącego odpowiednikiem UNIX AT&T V.3.2;
COMMAX TECH., INC.	– mikrokomputery EXCELL 486, 386, 286 i XT oraz laptopy;
WESTERN DIGITAL	– płyty główne 286, inteligentne dyski, karty sieciowe, karty graficzne, kontrolery dyskowe;
MAXTOR CORP.	– dyski o dużej pojemności i dyski optyczne;
CIPHER DATA PRODUCTS, INC.	– streamery, jednostki taśmowe 1/2";
DIGIBOARD	– inteligentne karty wielodostępu do mikrokomputerów;
OPTIMEM	– dyski optyczne;
SCO	– systemy UNIX i XENIX z Santa Cruz Operation.

Możemy także na żądanie klientów przedstawić oferty od innych producentów sprzętu i oprogramowania.

Uwaga!

Kompletacja dużych dostaw po rewelacyjnie niskich cenach

Motherboard 286, 12 MHz Clock
1 Mb RAM
Multi I/O
FDD/HDD Controller
1.2 Mb 5.25" FDD
360 Kb 5.25" FDD
40 Mb, 28 ms, HDD
MGP Card
14" Mono Monitor
Desktop Case
101 Key Keyboard

COMMAX

Już od 780 – USD. CIF Gdynia (min. 100 szt.).

Wszystkich zainteresowanych zapraszamy do naszego biura oraz do odwiedzenia naszego stoiska w pawilonie amerykańskim podczas tegorocznych Międzynarodowych Targów Poznańskich.

O/6/91

Skudlarski K.: Analizator Uogólnionych Stochastycznych Sieci Petriego INFORMATYKA 1991, nr 5, s. 3 Charakterystyka pakietu programów przeznaczonych do numerycznej analizy uogólnionych stochastycznych sieci Petriego na komputerach klasy IBM PC.	**Skudlarski K.: The program package for analysis of Generalized Stochastic Petri Nets** INFORMATYKA 1991, No. 5, p. 3 Characteristics of the program package for numeric analysis of generalized stochastic Petri Nets, which is adjusted for using on IBM PC computers.	**Skudlarski K.: Analysator für Verallgemeinerte Stochastische Petri-Netze** INFORMATYKA 1991, Nr. 5, S. 3 Eine Charakteristik des Programmpakets für numerische Analyse der verallgemeinerten stochastischen Petri-Netze auf Computern von IBM PC-Klasse.
Kuchta D.: Systematyczne testowanie programów – zalety i wady (2) INFORMATYKA 1991, nr 5, s. 8 Druga część artykułu na temat systematycznego testowania programów, zawierająca omówienie zalet i wad metod testowania przedstawionych w pierwszej części artykułu, a także propozycja ich usprawnienia pod kątem wzrostu skuteczności wykrywania błędów w programach.	**Kuchta D.: Systematic testing of programs – advantages and disadvantages (2)** INFORMATYKA 1991, No. 5, p. 8 Second part of the article on systematic program testing, which includes discussion of advantages and disadvantages of testing methods presented in the first part of the article, as well as proposals of their improving towards efficiency increase of error detection in computer programs.	**Kuchta D.: Systematische Testung von Programmen – die Vor- und Nachteile (2)** INFORMATYKA 1991, Nr. 5, S. 8 Zweiter Teil des Artikels über systematische Testung von Programmen, der eine Besprechung der Vor- und Nachteile von der im ersten Teil vorgestellten Testungsmethoden, sowie des Vorschlages einer Rationalisierung dieser Methoden in Richtung der Wirksamkeitssteigerung von Programmfehlererkennung umfasst.
Górecki A.: Komputeryzacja handlu detalicznego przy użyciu kas systemowych INFORMATYKA 1991, nr 5, s. 11 Charakterystyka współczesnych rozwiązań komputeryzacji handlu detalicznego oraz propozycje sposobu ich wprowadzenia w Polsce.	**Górecki A.: Retail trade computerization using system cash registers** INFORMATYKA 1991, No. 5, p. 11 Characteristics of contemporary solutions of retail trade computerization and proposals for their introducing in Poland.	**Górecki A.: Computerisierung des Einzelhandels mit Hilfe der Systemkassen** INFORMATYKA 1991, Nr 5, S. 11 Eine Charakteristik der zeitgenössischen Computerisierungslösungen im Einzelhandel und Vorschläge des Einsatzes solcher Lösungen in Polen.
Dołęga C.G.: Problemy wyszukiwania znaków w rozpoznawaniu tekstu drukowanego INFORMATYKA 1991, nr 5, s. 16 Charakterystyka algorytmów zwiększających efektywność automatycznego rozpoznawania tekstów drukowanych.	**Dołęga C.G.: Problems of character retrieval in printed text recognition** INFORMATYKA 1991, No. 5, p. 16 Characteristics of algorithms which increase effectivity of automatized recognition of printed text.	**Dołęga C.G.: Probleme des Zeichenaussuchens bei Erkennung des Drucktextes** INFORMATYKA 1991, Nr. 5, S. 16 Eine Charakteristik der Algorithmen für Effektivitätssteigerung der automatisierten Erkennung von Drucktexten.

Systematyczne testowanie programów ...

dokończenie ze s. 9

Niestety nie gwarantuje ono, że przetestowane programy będą dawały poprawne wyniki we wszystkich przypadkach występujących w praktyce. Nawet w przypadku najprostszych programów, jak przedstawione trywialne przykłady, zbiór dość łatwych do popełnienia błędów i ich skutków jest tak różnorodny, że testowanie nie gwarantuje wysokiego stopnia poprawności przetestowanych programów. Problem ten jest ważny, ponieważ skutki błędów mogą być kosztowne, a w niektórych dziedzinach bardzo niebezpieczne.

Oczywiście w praktyce trywialne błędy przedstawione w artykule zwykle nie są popełniane lub są wcześniej wykrywane, dzięki wiedzy, doświadczeniu i intuicji programistów oraz osób testujących programy. Nie można ich jednak generalnie wykluczyć, dopóki nie istnieją możliwe do praktycznego zastosowania metody weryfikacji, zapewniające ich znalezienie. Teoretyczna analiza błędów możliwych do popełnienia i ich wykrywalność przy stosowanych metodach testowania, taka jak ta przeprowadzona w niniejszym artykule, może przyczynić się do uwrażliwienia programisty na źródła błędów i pomóc testującym w wyborze najlepszych, czyli najbardziej wrażliwych przypadków testowych.

LITERATURA

[1] Gayen J.T., Kuchta D.: (1985–1989) Eigenschaften eingesetzter Nachweismethoden. Institut für Verkehr, Eisenbahnwesen und Verkehrssicherung, Technische Universitaet Braunschweig, BMFT Forschungsvorhaben, 1985–1989

[2] Gayen J.T., Kuchta D.: Possibilities and limitations of error detection by White-Box testing methods, including the domain borders method. Proc. of IFAC SAFECOMP'88, pp. 35–40, 1988

[3] Kuchta D.: Systematyczne testowanie programów – zalety i wady (1). INFORMATYKA, nr 4, 1991

[4] Myers G.: The Art of Software Testing. New York, 1978

[5] White L.J., Cohen E.I.: A domain strategy for computer program testing. IEEE Trans. Soft. Engin. No 5, pp. 247–257, 1980

Do końca maja przyjmujemy prenumeratę INFORMATYKI na drugie półrocze. Zaległe egzemplarze wysyłamy za zaliczeniem pocztowym.

Komputeryzacja handlu detalicznego ...

dokończenie ze s. 14

Zadania PNW obejmowałyby:

- przydzielanie kodów producenta,
- wydawanie skróconych numerów wyrobów,
- działalność doradcza w sprawie sposobu druku kodów,
- działalność doradcza w sprawie umieszczania kodów na wyrobach,
- rejestracja problemów wykrytych przez przedsiębiorstwa handlowe i przekazywanie ich producentom,
- działalność doradcza w sprawie zasad organizacji sprzedaży w sklepach wyposażonych w kasy systemowe,
- działalność na rzecz przekonania społeczeństwa do zaakceptowania idei sklepów, w których nie oznacza się cen na wyrobach,
- reprezentacja organizacji członkowskich w kontaktach z organizacjami rządowymi oraz środkami masowego przekazu.

Ze względu na obecny brak motywacji ekonomicznej, w początkowym okresie władze państwowe musiałyby udzielić finansowego poparcia dla tej inicjatywy.

Wprowadzenie kas systemowych do polskiego handlu jest nieuniknione i jest to tylko kwestią czasu. Dobrze byłoby ten czas racjonalnie wykorzystać. Pozwoliłoby to uniknąć problemów napotkanych przez te kraje, które już wcześniej wprowadziły tę nową technologię.

Problemy wyszukiwania znaków w rozpoznawaniu tekstu drukowanego

dokończenie ze s. 18

W przypadku pisma typu *kerning* poszczególne litery nie są sklejone, lecz przesłaniają się w taki sposób, iż funkcja $v(\ )$ nie osiąga w żadnym punkcie wartości zero. Uzyskanie poprawnego podziału wiersza na znaki za pomocą metod opartych na rzucie $v(\ )$, jest w takiej sytuacji dość skomplikowane.

LITERATURA

[1 Akiyama T., Hagita N., Masuda J., Naito S., Takahashi T.: Approach to Smart Document Reader System. Proceedings of IEEE Computer Society Conference „Computer Vision and Pattern Recognition", California 1985, p. 550

[2] Dersp V.: Scannen und Erkennen. C'T Magazin für Computer Technik, 4, 1990, S. 214, Heise Verlag GmbH.

FOR rok = 1 TO 5 Studiowanie NEXT rok

Ten felieton jest przeznaczony dla studentów informatyki

Brałem ostatnio udział w egzaminie dyplomowym. Dyplomant po omówieniu swojej pracy miał odpowiedzieć na dwa pytania z zakresu informatyki. Jedno z pytań dotyczyło programowania obiektowego. I tu nastąpiła konfuzja, brak odpowiedzi. Pytania ratunkowe dotyczyły już rozszerzeń w Turbo Pascalu 5.5 oraz istotnie nowych cech języka C++. Niestety również było brak odpowiedzi. Dyplomant jednak dostał dyplom, gdyż w jego programie studiów tematu „programowanie obiektowe" jeszcze nie było. Sytuacja taka nie jest wyjątkowa. Okazuje się, że większość studentów ogranicza się tylko do przyswojenia wiedzy ograniczonej programem studiów.

W poprzednich latach takie postępowanie nie prowadziło do większych perturbacji – wszak wystarczyło pokazać dyplom uczelni, aby być inżynierem. Co więcej, można było pokazać dowolny dyplom, aby być uznanym za informatyka. Spotykałem się z faktami wyjazdu na praktyki zagraniczne związane ze specjalistyczną informatyką, osób mających li tylko słabe pojęcie o programowaniu w Fortranie. Zaś ich przyjaciele pocieszali – jeżeli ktoś zdziwi się, że tak słabo znacie informatykę – zawsze możecie powiedzieć, że w Polsce tego nie uczyli.

Smutno było tego słuchać. Pomimo stałych ogromnych trudności w dostępie do nowoczesnych komputerów, układów i literatury – polskie uczelnie potrafiły jednak przygotować dobrych informatyków. Dowodem na to jest jakże liczna rzesza naszych koleżanek i kolegów pracująca z sukcesami na Zachodzie i jeszcze niedawno w Kuwejcie. Większość z nich, po chwilowym zachłyśnięciu się powszechnym dostępem do nowoczesnego sprzętu, szybko potrafiła stać się uznanymi specjalistami.

Musieli oni jednak spełnić wiele warunków. Po pierwsze dobrze opanować język angielski, zarówno w rozumieniu tekstów technicznych – jak i w mowie. Pożądana była też znajomość innego języka obcego. Musieli też być niezłymi studentami, nie ograniczającymi się wyłącznie do wiedzy skryptowej. O odwadze, a często konieczności podjęcia bardzo trudnych życiowych decyzji, nie ma co wspominać. Pozostali, nie spełniający tych warunków, mogli spokojnie egzystować na etatach w kraju.

Dlaczego teraz o tym piszę. Przecież w zasadzie to wszyscy wiedzą. Tak, ale jest to felieton dla obecnych studentów, dla których nie będzie już po skończeniu przez nich studiów taryfy ulgowej. Zachód dla informatyków przychodzi do nas, do Polski. W niedługim czasie szanse na dobrą pracę i płace tutaj w kraju będą mieli tylko ci, którzy będą mieli rzeczywiste umiejętności, a nie tylko dyplom. Również znajomość języka angielskiego, niemieckiego i francuskiego będzie się liczyć. Coraz więcej firm zachodnich otwiera lub zamierza utworzyć w Polsce swoje oddziały. Potrzebni będą im profesjonalni (a nie tylko przyuczeni) informatycy. Również wiele polskich firm komputerowych będzie mogło wejść w kooperację z firmami zachodnimi, ale też tylko pod warunkiem, że będą dysponować dobrą kadrą.

Przy jeszcze większych niż poprzednio trudnościach sprzętowych i kadrowych uczelnie starają się nauczyć podstaw wiedzy informatycznej. Przy tym nie jest istotne, że nie uczy się powszechnie najnowocześniejszych języków programowania, systemu UNIX czy określonej bazy danych. Często brak sprzętu ogranicza praktykę programowania do języków Turbo Pascal i C oraz systemów MS-DOS i dBASE. Dopiero na trzecim czy czwartym roku student ma możliwość dotknięcia klawiatury mikrokomputera. To prawda, ale umiejętności informatyczne nie rodzą się przy klawiaturze. Najpierw należy nauczyć się podstaw konstruowania algorytmów, abstrakcyjnych typów danych, czy struktury statycznej i dynamicznej systemu operacyjnego oraz elementów klasycznych operacji synchronizacji. Przecież teraz trudno przewidzieć, jakiego systemu i języka programowania będzie używał za parę lat obecny student z trzeciego roku. Wtedy też, będzie mu potrzebne szybkie nabycie praktycznych umiejętności posługiwania się konkretnym sprzętem i oprogramowaniem. I będzie mógł podołać temu zadaniu tylko wtedy, jeżeli będzie miał odpowiednie podstawy teoretyczne. Dlatego w moim pojęciu nie jest nawet tak bardzo istotny brak nowoczesnego sprzętu na uczelniach. Inna jest sprawa, że dostęp do takiego sprzętu znacznie by uatrakcyjnił studia oraz wzmocnił naukę teorii informatyki możliwie najnowszą praktyką.

Wróćmy jednak do apelu skierowanego do przyszłych absolwentów informatyki. Uważam, że będziecie dalej nieźle przygotowani teoretycznie z podstawowej wiedzy informatycznej (jeżeli dalsza pauperyzacja kadry uczelnianej nie spowoduje samolikwidacji instytutów). Musicie jednak stale się doszkalać i poszukiwać samodzielnie informacji o nowych rozwiązaniach. Po skończeniu uczelni, zależnie od rodzaju pracy, będziecie musieli się dokształcać na specjalistycznych kursach (często płacąc za nie osobiście). Do podjęcia działalności zawodowej i pełnego uczestnictwa w kursach konieczna będzie znajomość języków obcych. Dopiero wtedy będziecie mogli marzyć o dostaniu dobrej pracy i płacy.

Uważam też, że wiele istniejących i nowo powstających zespołów będzie musiało oprzeć się na młodszych wiekiem informatykach. Bowiem, w moim przekonaniu, tylko oni nadają się do podjęcia wytężonej pracy nad nowymi systemami. Mają lepsze przygotowanie teoretyczne, świeższy umysł i więcej zapału. Musi być jednak spełniony jeden warunek. Nad ich pracą muszą czuwać doświadczeni liderzy projektów, aby nadmiar zapału nie powodował powstawania superproduktów, nigdy rzetelnie nie przetestowanych. Problemem jest brak u nas w kraju takich liderów. Gdzie mieli oni zdobywać doświadczenie, gdy w ostatnich latach najpoważniejsze systemy powstawały w oparciu o dBASE, a w Kuwejcie pracowali jako biali Murzyni.

dokończenie na s. 26

PRZEDSIĘBIORSTWO
ZASTOSOWAŃ INFORMATYKI
meditronik
oferuje

UM7106	18 000
UM3481,2,3,4,10,11	8200
UM66T	3600
UM82450	14 000
UM8250B	14 000
UM8253-5	16 500
UM8259A-2	27 300
UM82C11	13 000
UM82C55A	16 500
UM82C8167	19 500
UM6116U-35(ns)	25 500
UM6116-2L	11 500
UM6264-10	20 000
UM62256A-10L	52 000

Nasz adres:
PZI „MEDITRONIK"
00-194 Warszawa, ul. Dzika 4
telefon 635-22-63, 635-22-64
telefaks 635-21-95
teleks 816075 medi pl
0/24/90

Przedsiębiorstwo Przemysłu Mięsnego
54-203 Wrocław,ul. Legnicka 58
tel. 56-44-76, teleks 712441, faks 55-46-79
odsprzeda w całości lub w elementach

zestaw komputerowy oparty na minikomputerze KRS 4201 firmy ROBOTRON,

w skład którego wchodzą m.in.:

- jednostki dyskowe IZOT 1370 – 4 szt.
- przewijarki taśmowe IZOT 5003 – 8 szt.
- czytniki nośników papierowych

oraz

taśmy magnetyczne i dyski do ww. urządzeń.

0/8/91